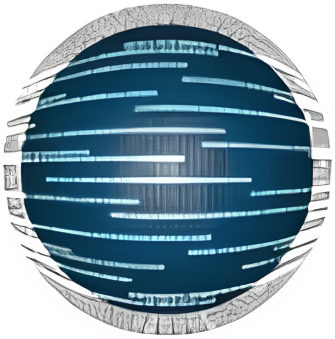


Stock Prediction with Linear Regression for NFLX

Στατιστικές Μέθοδοι Μηχανικής Μάθησης

Βασίλειος Μπίτζας 1083796 up1083796@ac.upatras.gr

Τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστών & Πληροφορικής
Πανεπιστήμιο Πατρών

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

1.1 Εισαγωγή

Η πρόβλεψη τιμών μετοχών (stock price prediction) αποτελεί ένα από τα πιο προκλητικά προβλήματα στον τομέα της μηχανικής μάθησης (machine learning) και της χρηματοοικονομικής ανάλυσης (financial analysis). Η πολυπλοκότητα των χρηματοοικονομικών αγορών, η επίδραση εξωγενών παραγόντων και η μη-γραμμική φύση των χρονοσειρών (time series) καθιστούν την ακριβή πρόβλεψη μια σύνθετη διαδικασία.

Στην παρούσα εργασία, αναπτύσσεται ένα ολοκληρωμένο σύστημα πρόβλεψης για τις τιμές της μετοχής της Netflix (NFLX), χρησιμοποιώντας στατιστικές μεθόδους μηχανικής μάθησης [Hastie, James 2009, 2013]. Συγκεκριμένα, εφαρμόζουμε τεχνικές γραμμικής παλινδρόμησης (linear regression), πολυωνυμικής παλινδρόμησης (polynomial regression) με κανονικοποίηση (regularization), και μεθόδους μείωσης διαστάσεων (dimensionality reduction).

1.2 Στόχοι Εργασίας

Οι κύριοι στόχοι της παρούσας εργασίας είναι:

1. **Εργασία Α**: Ανάπτυξη baseline μοντέλων γραμμικής παλινδρόμησης για την εύρεση της σχέσης μεταξύ παρελθοντικών τιμών κλεισίματος (closing prices) και της επόμενης τιμής.
2. **Εργασία Β**: Υλοποίηση πολυωνυμικής παλινδρόμησης με L1 (Lasso) και L2 (Ridge) κανονικοποίηση για τη σύλληψη μη-γραμμικών σχέσεων (non-linear relationships).
3. **Εργασία Γ**: Εφαρμογή τεχνικών μείωσης διαστάσεων (PCA, CFS, Sequential Forward Selection) για τη βελτιστοποίηση των χαρακτηριστικών (feature optimization).
4. **Εργασία Δ**: Πρόβλεψη μελλοντικών τιμών για Δεκέμβριο 2025 και Ιανουάριο 2026, με αξιολόγηση της αξιοπιστίας των προβλέψεων.

1.3 Περιγραφή Δεδομένων

Τα δεδομένα που χρησιμοποιούνται στην παρούσα εργασία προέρχονται από το Alpha Vantage API [Alpha Vantage Inc. 2024] και περιλαμβάνουν:

- **Σύμβολο Μετοχής**: NFLX (Netflix, Inc.)
- **Τομέας**: Communication Services
- **Χρονική Περίοδος**: Μάιος 2002 — Νοέμβριος 2025
- **Συνολικοί Μήνες**: 283 μηνιαία δεδομένα
- **Χαρακτηριστικά**: Τιμή κλεισίματος (close price), Όγκος συναλλαγών (trading volume)

Τα ημερήσια δεδομένα μετατράπηκαν σε μηνιαίους μέσους όρους για τη μείωση του θορύβου (noise reduction) και την καλύτερη ανίχνευση μακροχρόνιων τάσεων (long-term trends).

Σημαντική Σημείωση για Δωρεάν API Keys / Important Note for Free API Keys

Ελληνικά: Η παράμετρος `outputsize=full` απαιτεί premium (επί πληρωμή) API key. Για δωρεάν API keys, χρησιμοποιήστε **`outputsize=compact`** αντί για `outputsize=full`. Το `compact` επιστρέφει τα τελευταία 100 data points (~3–4 μήνες ημερήσιων δεδομένων).
English: The parameter `outputsize=full` requires a premium (paid) API key. For free API keys, use **`outputsize=compact`** instead of `outputsize=full`. The `compact` option returns the latest 100 data points (~3–4 months of daily data).
Τροποποίηση Κώδικα / Code Modification:

```
# Αλλάξτε από (Change from):
outputsize=full
# Σε (To):
outputsize=compact
```

Σημείωση για το Notebook: Το `nflx_stock_prediction_complete_pipeline.ipynb` έχει ήδη εκτελεστεί με πλήρη δεδομένα 20+ ετών (`outputsize=full`). Τα output cells δείχνουν αποτελέσματα από αυτήν την εκπαίδευση.
Note about Notebook: The `nflx_stock_prediction_complete_pipeline.ipynb` has already been executed with full 20+ years of data (`outputsize=full`). The output cells show results from this training.

Η παράμετρος `outputsize=full` επιτρέπει την ανάκτηση 20+ ετών ιστορικών δεδομένων (μόνο για premium keys).

1.4 Διάρθωση Αναφοράς

Η αναφορά οργανώνεται ως εξής:

- **Κεφάλαιο 2**: Παρουσιάζεται το θεωρητικό υπόβαθρο των μεθόδων που χρησιμοποιούνται, με μαθηματικές παραγωγές και θεωρητική ανάλυση.
- **Κεφάλαιο 3**: Περιγράφεται η μεθοδολογία που ακολουθήθηκε, από τη συλλογή δεδομένων έως την εκπαίδευση των μοντέλων.
- **Κεφάλαιο 4**: Παρουσιάζονται λεπτομέρειες υλοποίησης, αρχιτεκτονική κώδικα και τεχνικές επιλογές.
- **Κεφάλαιο 5**: Αναλύονται τα αποτελέσματα των 96 μοντέλων που εκπαιδεύτηκαν, με εκτενείς οπτικοποιήσεις.
- **Κεφάλαιο 6**: Συζητούνται τα ευρήματα, οι περιορισμοί και οι πρακτικές εφαρμογές.
- **Κεφάλαιο 7**: Παρουσιάζονται τα συμπεράσματα και προτάσεις για μελλοντική έρευνα.

Κεφάλαιο 2

Θεωρητικό Υπόβαθρο

2.1 Γραμμική Παλινδρόμηση

2.1.1 Μαθηματικό Υπόδειγμα

Η γραμμική παλινδρόμηση (linear regression) αποτελεί τη θεμελιώδη μέθοδο για τη μοντελοποίηση της σχέσης μεταξύ μιας εξαρτημένης μεταβλητής y και μιας ή περισσότερων ανεξάρτητων μεταβλητών $\mathbf{x}$. Το γενικό μοντέλο εκφράζεται ως:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon \tag{2.1}$$

όπου:

- y: η εξαρτημένη μεταβλητή (target variable)
- $x_1, x_2, \ldots, x_p$: οι ανεξάρτητες μεταβλητές (features)
- β_0: η σταθερά (intercept)
- $\beta_1, \beta_2, \ldots, \beta_p$: οι συντελεστές (coefficients)
- ϵ: το σφάλμα (error term) με $\epsilon \sim \mathcal{N}(0, \sigma^2)$

Σε διανυσματική μορφή:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{2.2}$$

όπου $\mathbf{y} \in \mathbb{R}^n$, $\mathbf{X} \in \mathbb{R}^{n \times (p+1)}$, $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$, και $\boldsymbol{\epsilon} \in \mathbb{R}^n$.

2.1.2 Μέθοδος Ελαχίστων Τετραγώνων

Η εκτίμηση των παραμέτρων $\boldsymbol{\beta}$ γίνεται με τη μέθοδο των ελαχίστων τετραγώνων (Ordinary Least Squares - OLS), η οποία ελαχιστοποιεί το άθροισμα των τετραγώνων των υπολοίπων (residual sum of squares):

$$\text{RSS}(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \tag{2.3}$$

Για την εύρεση του ελαχίστου, παραγωγίζουμε ως προς $\boldsymbol{\beta}$ και θέτουμε ίσο με μηδέν:

$$\frac{\partial \text{RSS}}{\partial \boldsymbol{\beta}} = -2\mathbf{X}^T(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) = 0 \tag{2.4}$$

Λύνοντας ως προς $\boldsymbol{\beta}$, προκύπτει η κλειστή λύση:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{2.5}$$

2.1.3 Παραδοχές και Περιορισμοί

Η γραμμική παλινδρόμηση βασίζεται στις ακόλουθες παραδοχές (Gauss-Markov assumptions):

1. **Γραμμικότητα**: Η σχέση μεταξύ $\mathbf{X}$ και y είναι γραμμική.
2. **Ανεξαρτησία**: Οι παρατηρήσεις είναι ανεξάρτητες μεταξύ τους.
3. **Ομοσκεδαστικότητα**: Η διακύμανση του σφάλματος είναι σταθερή, $\text{Var}(\epsilon_i) = \sigma^2$.
4. **Κανονικότητα**: Τα σφάλματα ακολουθούν κανονική κατανομή, $\epsilon \sim \mathcal{N}(0, \sigma^2)$.
5. **Μη-πολυσυγγραμμικότητα**: Οι ανεξάρτητες μεταβλητές δεν έχουν γραμμική εξάρτηση.

2.2 Πολυωνυμική Παλινδρόμηση

2.2.1 Επέκταση Γραμμικού Μοντέλου

Η πολυωνυμική παλινδρόμηση (polynomial regression) επεκτείνει το γραμμικό μοντέλο για να συλλάβει μη-γραμμικές σχέσεις. Για μία μεταβλητή x και βαθμό d:

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_d x^d + \epsilon \tag{2.6}$$

Για πολλές μεταβλητές, το πολυωνυμικό μοντέλο βαθμού 2 (quadratic) περιλαμβάνει:

$$y = \beta_0 + \sum_{j=1}^{p} \beta_j x_j + \sum_{j=1}^{p} \sum_{k=j}^{p} \beta_{jk} x_j x_k + \epsilon \tag{2.7}$$

2.2.2 Πολυωνυμικά Χαρακτηριστικά

Ο μετασχηματισμός πολυωνυμικών χαρακτηριστικών (polynomial features) δημιουργεί νέες μεταβλητές:

$$\phi : \mathbb{R}^p \to \mathbb{R}^P, \quad \mathbf{x} \mapsto [1, x_1, \ldots, x_p, x_1^2, x_1 x_2, \ldots, x_p^d] \tag{2.8}$$

όπου $P = \binom{p+d}{d}$ για βαθμό d.

2.2.3 Κίνδυνος Υπερπροσαρμογής

Η αύξηση του βαθμού του πολυωνύμου αυξάνει την πολυπλοκότητα του μοντέλου (model complexity), οδηγώντας σε κίνδυνο υπερπροσαρμογής (overfitting). Το trade-off μεταξύ bias και variance εκφράζεται ως:

$$\text{Expected MSE} = \text{Bias}^2 + \text{Variance} + \text{Irreducible Error} \tag{2.9}$$

2.3 Κανονικοποίηση

2.3.1 L2 Regularization (Ridge)

Η Ridge regression [Hoerl 1970] προσθέτει έναν όρο ποινής (penalty term) στην αντικειμενική συνάρτηση:

$$\text{RSS}_{\text{Ridge}}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \alpha\|\boldsymbol{\beta}\|^2 \tag{2.10}$$

όπου $\alpha \geq 0$ είναι η παράμετρος κανονικοποίησης (regularization parameter). Η λύση είναι:

$$\hat{\boldsymbol{\beta}}_{\text{Ridge}} = (\mathbf{X}^T\mathbf{X} + \alpha\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y} \tag{2.11}$$

Η Ridge regression μειώνει το μέγεθος των συντελεστών (coefficient shrinkage), βελτιώνοντας τη γενίκευση όταν υπάρχει πολυσυγγραμμικότητα (multicollinearity).

2.3.2 L1 Regularization (Lasso)

Η Lasso regression [Tibshirani 1996] χρησιμοποιεί L1 κανονικοποίηση:

$$\text{RSS}_{\text{Lasso}}(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 + \alpha\|\boldsymbol{\beta}\|_1 \tag{2.12}$$

όπου $\|\boldsymbol{\beta}\|_1 = \sum_{j=1}^{p} |\beta_j|$. Η Lasso παράγει αραιά μοντέλα (sparse models), θέτοντας ορισμένους συντελεστές ακριβώς στο μηδέν, επιτελώντας έτσι αυτόματη επιλογή χαρακτηριστικών (automatic feature selection).

2.3.3 Σύγκριση L1 vs L2

Χαρακτηριστικό	**L1 (Lasso)**	**L2 (Ridge)**
Penalty Term	$\alpha \sum \lvert\beta_j\rvert$	$\alpha \sum \beta_j^2$
Λύση	Numerical (iterative)	Closed-form (Eq. 2.11)
Αραιότητα (Sparsity)	Ναι ($\beta_j = 0$)	Όχι ($\beta_j \approx 0$)
Feature Selection	Αυτόματη	Όχι
Multicollinearity	Επιλέγει 1 feature	Κατανέμει βάρη

Πίνακας 2.1: Σύγκριση L1 και L2 κανονικοποίησης

2.4 Μείωση Διαστάσεων

2.4.1 Principal Component Analysis (PCA)

Το PCA [Jolliffe 2002] είναι μη-εποπτευόμενη μέθοδος (unsupervised method) που μετασχηματίζει τα δεδομένα σε νέο σύστημα συντεταγμένων με μέγιστη διακύμανση (maximum variance).

Μαθηματική Παραγωγή:

Έστω $\mathbf{X} \in \mathbb{R}^{n \times p}$ ο κεντραρισμένος πίνακας δεδομένων. Ο πίνακας συνδιακύμανσης (covariance matrix) είναι:

$$\mathbf{C} = \frac{1}{n-1}\mathbf{X}^T\mathbf{X} \tag{2.13}$$

Το PCA βρίσκει τα ιδιοδιανύσματα (eigenvectors) και ιδιοτιμές (eigenvalues) του $\mathbf{C}$:

$$\mathbf{C}\mathbf{v}_i = \lambda_i \mathbf{v}_i \tag{2.14}$$

Οι κύριες συνιστώσες (principal components) είναι:

$$\mathbf{Z} = \mathbf{X}\mathbf{V} \tag{2.15}$$

όπου $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k]$ περιέχει τα πρώτα k ιδιοδιανύσματα. Η διακύμανση που εξηγείται (explained variance) είναι:

$$\text{Explained Variance Ratio} = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{p} \lambda_i} \tag{2.16}$$

2.4.2 Correlation-based Feature Selection (CFS)

Το CFS [Hall 1999] είναι filter method που αξιολογεί υποσύνολα χαρακτηριστικών βάσει της συσχέτισης τους με τον στόχο και μεταξύ τους. Το merit score ορίζεται ως:

$$\text{Merit}_S = \frac{k\overline{r_{cf}}}{\sqrt{k + k(k-1)\overline{r_{ff}}}} \tag{2.17}$$

όπου:

- k: αριθμός χαρακτηριστικών
- $\overline{r_{cf}}$: μέση συσχέτιση χαρακτηριστικών-κλάσης
- $\overline{r_{ff}}$: μέση συσχέτιση χαρακτηριστικών-χαρακτηριστικών

2.4.3 Sequential Forward Selection

Το SFS [Guyon 2003] είναι wrapper method που επιλέγει χαρακτηριστικά greedy. Ο αλγόριθμος λειτουργεί ως εξής:

1. Ξεκινάμε με κενό σύνολο χαρακτηριστικών $S = \emptyset$
2. Για k επαναλήψεις:
   (α΄) Αξιολογούμε όλα τα χαρακτηριστικά $x \in F \setminus S$
   (β΄) Επιλέγουμε $x^* = \arg\max_{x \in F \setminus S} \text{Score}(S \cup \{x\})$
   (γ΄) Προσθέτουμε $S \leftarrow S \cup \{x^*\}$
3. Επιστρέφουμε το σύνολο S

όπου το Score μπορεί να είναι cross-validated R^2 ή άλλη μετρική απόδοσης.

2.5 Μετρικές Αξιολόγησης

2.5.1 Root Mean Squared Error (RMSE)

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2.18}$$

Το RMSE εκφράζει το μέσο σφάλμα πρόβλεψης στις ίδιες μονάδες με τη μεταβλητή-στόχο.

2.5.2 Mean Absolute Error (MAE)

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{2.19}$$

Το MAE είναι λιγότερο ευαίσθητο σε outliers από το RMSE.

2.5.3 Coefficient of Determination (R^2)

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{\text{RSS}}{\text{TSS}} \tag{2.20}$$

όπου TSS (Total Sum of Squares) = $\sum_{i=1}^{n}(y_i - \bar{y})^2$. Το $R^2 \in [0, 1]$ εκφράζει το ποσοστό της διακύμανσης που εξηγείται από το μοντέλο.

2.5.4 Διαστήματα Εμπιστοσύνης

Για γραμμική παλινδρόμηση, το $(1 - \alpha)$ διάστημα εμπιστοσύνης (confidence interval) για πρόβλεψη είναι:

$$\hat{y} \pm t_{\alpha/2, n-p-1} \cdot \text{SE}(\hat{y}) \tag{2.21}$$

όπου:

$$\text{SE}(\hat{y}) = \hat{\sigma}\sqrt{1 + \mathbf{x}_0^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{x}_0} \tag{2.22}$$

και $\hat{\sigma}^2 = \frac{1}{n-p-1}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$.

2.6 Χρονοσειρές και Lagged Features

2.6.1 Χαρακτηριστικά Καθυστέρησης

Για χρονοσειρές, τα lagged features δημιουργούν ένα autoregressive μοντέλο:

$$y_t = \beta_0 + \beta_1 y_{t-1} + \beta_2 y_{t-2} + \cdots + \beta_p y_{t-p} + \epsilon_t \tag{2.23}$$

Αυτό είναι γνωστό ως AR(p) (Autoregressive model of order p).

2.6.2 Χρονολογική Διαίρεση Δεδομένων

Για χρονοσειρές, η διαίρεση πρέπει να διατηρεί τη χρονική σειρά:

$$\begin{aligned} \text{Training} &= \{(x_t, y_t) : t \in [1, T_{\text{train}}]\} \\ \text{Validation} &= \{(x_t, y_t) : t \in [T_{\text{train}}, T]\} \end{aligned} \tag{2.24}$$

Τυχαία ανακάτεμα (random shuffling) οδηγεί σε data leakage και υπερεκτίμηση της απόδοσης.

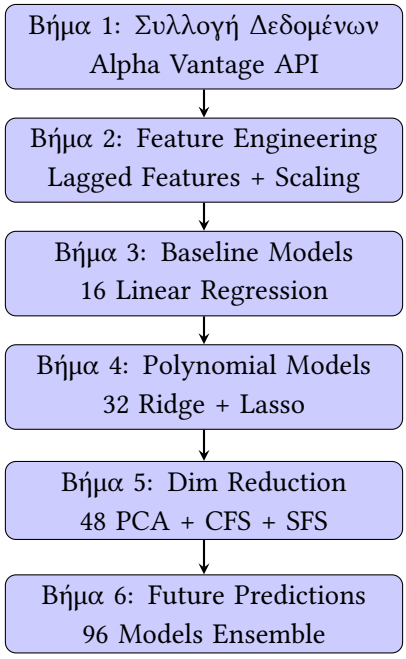
Κεφάλαιο 3

Μεθοδολογία

3.1 Επισκόπηση Pipeline

3.1.1 Αρχιτεκτονική Συστήματος

Το σύστημα πρόβλεψης τιμών μετοχών NFLX αναπτύχθηκε ως ένα modular pipeline αποτελούμενο από έξι διακριτά στάδια (stages), όπως απεικονίζεται στο Σχήμα 3.1. Κάθε στάδιο υλοποιείται ως ανεξάρτητο Python script, επιτρέποντας την ευκολία επαναχρησιμοποίησης (reusability) και συντήρησης (maintainability).



Σχήμα 3.1: Αρχιτεκτονική pipeline για πρόβλεψη τιμών NFLX

3.1.2 Ροή Εργασίας

Η ροή εργασίας (workflow) περιλαμβάνει τα ακόλουθα στάδια:

1. **Data Acquisition**: Λήψη ημερήσιων δεδομένων από Alpha Vantage API
2. **Preprocessing**: Μετατροπή σε μηνιαία δεδομένα, Gaussian smoothing
3. **Feature Engineering**: Δημιουργία lagged features, normalization
4. **Model Training**: Εκπαίδευση 96 μοντέλων με διαφορετικές προσεγγίσεις
5. **Validation**: Αξιολόγηση σε validation set 2025
6. **Prediction**: Πρόβλεψη για Δεκέμβριο 2025 και Ιανουάριο 2026

3.1.3 Τεχνολογίες και Εργαλεία

Το έργο υλοποιήθηκε χρησιμοποιώντας τις ακόλουθες βιβλιοθήκες Python:

- `pandas (v1.5+)` [The pandas development team 2020]: Χειρισμός και ανάλυση δεδομένων
- `numpy (v1.23+)` [Harris 2020]: Αριθμητικοί υπολογισμοί και linear algebra
- `scikit-learn (v1.2+)` [Pedregosa 2011]: Machine learning αλγόριθμοι και μετρικές
- `scipy (v1.10+)`: Επιστημονικοί υπολογισμοί, Gaussian filtering
- `matplotlib (v3.6+)` [Hunter 2007]: Οπτικοποίηση δεδομένων και αποτελεσμάτων
- `requests (v2.28+)`: HTTP requests για API communication

3.2 Βήμα 1: Συλλογή και Προεπεξεργασία Δεδομένων

3.2.1 Alpha Vantage API

Η συλλογή δεδομένων πραγματοποιήθηκε μέσω του Alpha Vantage API [Alpha Vantage Inc. 2024], το οποίο παρέχει δωρεάν πρόσβαση σε ιστορικά χρηματοοικονομικά δεδομένα. Το API endpoint που χρησιμοποιήθηκε είναι:

```
https://www.alphavantage.co/query?
    function=TIME_SERIES_DAILY&
    symbol=NFLX&
    outputsize=full&
    apikey=<API_KEY>
```

Σημαντική Σημείωση για Δωρεάν API Keys / Important Note for Free API Keys

Ελληνικά: Η παράμετρος `outputsize=full` απαιτεί premium (επί πληρωμή) API key. Για δωρεάν API keys, χρησιμοποιήστε **`outputsize=compact`** αντί για `outputsize=full`. Το `compact` επιστρέφει τα τελευταία 100 data points (~3–4 μήνες ημερήσιων δεδομένων).
English: The parameter `outputsize=full` requires a premium (paid) API key. For free API keys, use **`outputsize=compact`** instead of `outputsize=full`. The `compact` option returns the latest 100 data points (~3–4 months of daily data).
Τροποποίηση Κώδικα / Code Modification:

```
# Αλλάξτε από (Change from):
outputsize=full
# Σε (To):
outputsize=compact
```

Σημείωση για το Notebook: Το `nflx_stock_prediction_complete_pipeline.ipynb` έχει ήδη εκτελεστεί με πλήρη δεδομένα 20+ ετών (`outputsize=full`). Τα output cells δείχνουν αποτελέσματα από αυτήν την εκπαίδευση.
Note about Notebook: The `nflx_stock_prediction_complete_pipeline.ipynb` has already been executed with full 20+ years of data (`outputsize=full`). The output cells show results from this training.

Η παράμετρος `outputsize=full` επιτρέπει την ανάκτηση 20+ ετών ιστορικών δεδομένων (μόνο για premium keys).

3.2.2 Μετατροπή σε Μηνιαία Δεδομένα

Τα ημερήσια δεδομένα μετατράπηκαν σε μηνιαίους μέσους όρους για τη μείωση του θορύβου και την καλύτερη σύλληψη μακροχρόνιων τάσεων. Ο μετασχηματισμός υπολογίζεται ως:

$$\text{Close}_{\text{monthly}} = \frac{1}{N_d} \sum_{d=1}^{N_d} \text{Close}_d, \quad \text{Volume}_{\text{monthly}} = \frac{1}{N_d} \sum_{d=1}^{N_d} \text{Volume}_d \tag{3.1}$$

όπου N_d είναι ο αριθμός ημερών στον μήνα.

3.2.3 Gaussian Smoothing

Για την περαιτέρω μείωση θορύβου, εφαρμόστηκε Gaussian filtering [Lindeberg 1994] με τέσσερα επίπεδα smoothing:

$$y_{\text{smooth}}[n] = \sum_{k=-\infty}^{\infty} y[k] \cdot G(n-k;\sigma) \tag{3.2}$$

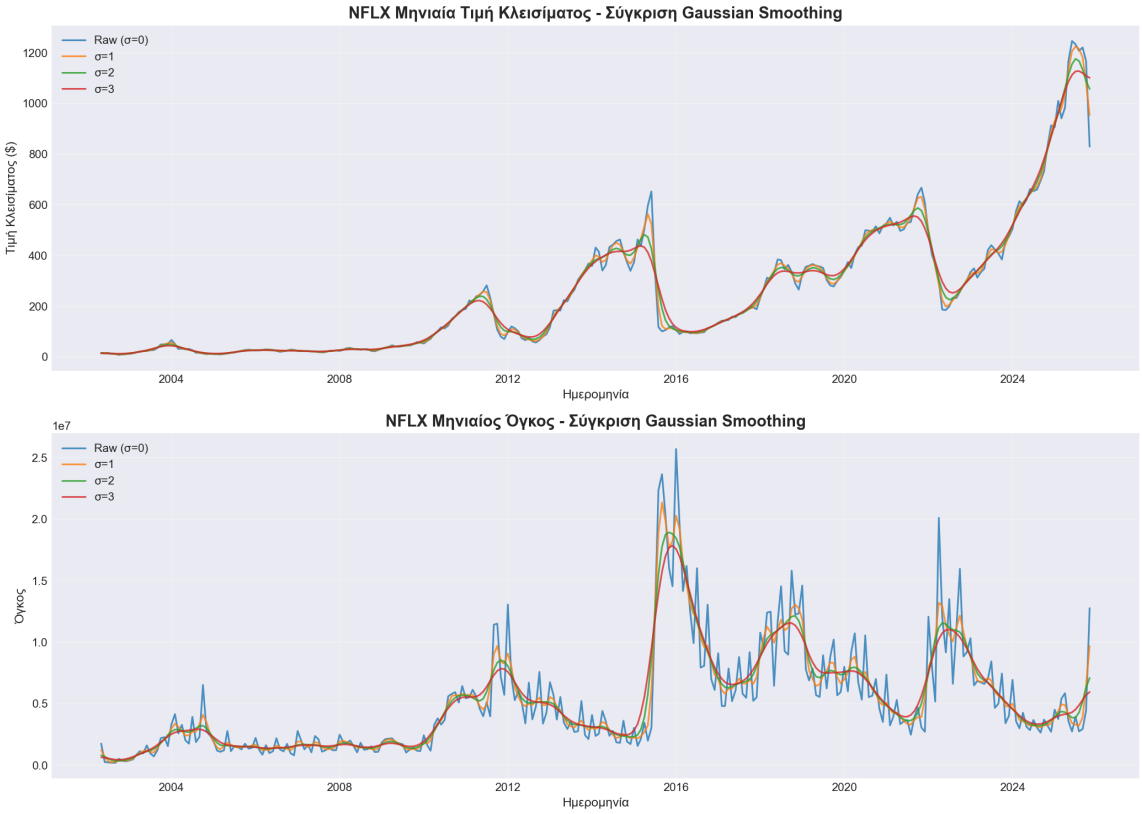
όπου $G(x;\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}$ είναι το Gaussian kernel και $\sigma \in \{0, 1, 2, 3\}$.

Δημιουργήθηκαν τέσσερα datasets:

- `raw`: Χωρίς smoothing ($\sigma = 0$)
- `sigma1`: Ελαφρύ smoothing ($\sigma = 1$)
- `sigma2`: Μέτριο smoothing ($\sigma = 2$)
- `sigma3`: Έντονο smoothing ($\sigma = 3$)

3.2.4 Οπτικοποίηση Δεδομένων

Το Σχήμα 3.2 δείχνει τη σύγκριση των τεσσάρων επιπέδων smoothing στα μηνιαία δεδομένα NFLX.



Σχήμα 3.2: Σύγκριση Gaussian smoothing με διαφορετικές τιμές σ

Παρατηρείται ότι το smoothing με $\sigma = 3$ μειώνει σημαντικά τις βραχυπρόθεσμες διακυμάνσεις διατηρώντας τη μακροχρόνια τάση.

3.3 Βήμα 2: Δημιουργία Χαρακτηριστικών

3.3.1 Lagged Features

Για κάθε επίπεδο smoothing, δημιουργήθηκαν lagged features για τέσσερα διαφορετικά time windows:

$$\mathbf{x}_t = [y_{t-1}, y_{t-2}, \ldots, y_{t-p}, v_{t-1}, v_{t-2}, \ldots, v_{t-p}]^T \tag{3.3}$$

όπου y είναι η τιμή κλεισίματος (close price), v είναι ο όγκος συναλλαγών (volume), και $p \in \{3, 6, 9, 12\}$ μήνες.

Για παράδειγμα, με $p = 12$, το feature vector έχει διάσταση $2 \times 12 = 24$ χαρακτηριστικά.

3.3.2 Χρονολογική Διαίρεση

Η διαίρεση των δεδομένων έγινε χρονολογικά για την αποφυγή data leakage:

- **Training Set**: Όλες οι παρατηρήσεις πριν το 2025 ($t < 2025$)
- **Validation Set**: Παρατηρήσεις του έτους 2025 ($t = 2025$)

Αυτό διασφαλίζει ότι το μοντέλο δεν έχει πρόσβαση σε μελλοντικές πληροφορίες κατά την εκπαίδευση.

3.3.3 StandardScaler Normalization

Κάθε feature normalization έγινε χρησιμοποιώντας StandardScaler:

$$x_{\text{scaled}} = \frac{x - \mu_{\text{train}}}{\sigma_{\text{train}}} \tag{3.4}$$

όπου μ_{train} και σ_{train} υπολογίζονται **μόνο** από το training set. Η ίδια μετατροπή εφαρμόζεται στο validation set για να αποφευχθεί data leakage.

3.3.4 Feature Matrix Construction

Συνολικά δημιουργήθηκαν 16 feature configurations:

Smoothing	Lags	Features	Filename
raw	3	6	`features_raw_3lags.npz`
raw	6	12	`features_raw_6lags.npz`
raw	9	18	`features_raw_9lags.npz`
raw	12	24	`features_raw_12lags.npz`
sigma1	3	6	`features_sigma1_3lags.npz`
⋮	⋮	⋮	⋮
sigma3	12	24	`features_sigma3_12lags.npz`

Πίνακας 3.1: Feature configurations που δημιουργήθηκαν

3.4 Βήμα 3: Baseline Linear Regression (Εργασία Α)

3.4.1 Εκπαίδευση 16 Μοντέλων

Για την Εργασία Α, εκπαιδεύτηκαν 16 baseline μοντέλα γραμμικής παλινδρόμησης, ένα για κάθε feature configuration. Κάθε μοντέλο εφαρμόζει την OLS λύση (Εξίσωση 2.5):

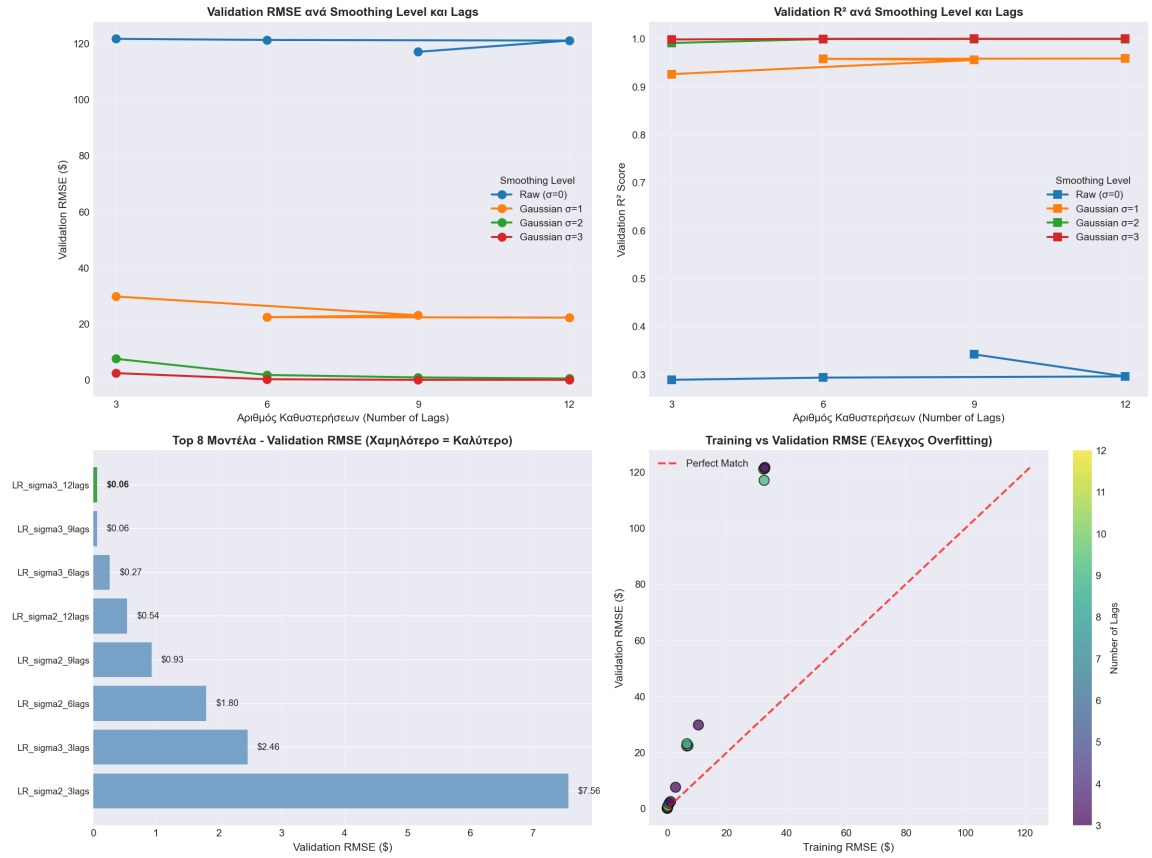
Baseline Linear Regression Training

```python
from sklearn.linear_model import LinearRegression

model = LinearRegression()
model.fit(X_train, y_train)
y_pred_val = model.predict(X_val)
```

3.4.2 Σύγκριση Ρυθμίσεων

Το Σχήμα 3.3 παρουσιάζει τη σύγκριση των 16 baseline μοντέλων βάσει validation RMSE.



Σχήμα 3.3: Σύγκριση απόδοσης 16 baseline μοντέλων

Παρατηρείται ότι τα μοντέλα με $\sigma = 3$ και $p = 12$ lags επιτυγχάνουν την καλύτερη απόδοση.

3.4.3 Ανάλυση Συντελεστών

Για το καλύτερο baseline μοντέλο (`LR_sigma3_12lags`), οι συντελεστές αποκαλύπτουν τη σημασία κάθε lagged feature. Οι πρώτες υστερήσεις ($t-1, t-2$) έχουν τα μεγαλύτερα βάρη, υποδεικνύοντας ισχυρή αυτοσυσχέτιση (autocorrelation).

3.5 Βήμα 4: Polynomial Regression (Εργασία Β)

3.5.1 Πολυωνυμικά Χαρακτηριστικά Βαθμού 2

Για την Εργασία Β, επεκτάθηκαν όλα τα 16 baseline configurations σε πολυωνυμικά χαρακτηριστικά βαθμού 2:

Polynomial Feature Transformation

```
from sklearn.preprocessing import PolynomialFeatures

poly = PolynomialFeatures(degree=2, include_bias=False)
X_train_poly = poly.fit_transform(X_train)
X_val_poly = poly.transform(X_val)
```

Για $p = 12$ lags (24 features), το πολυωνυμικό transformation δημιουργεί:

$$P = \frac{24 \times (24 + 1)}{2} = 300 \text{ features} \tag{3.5}$$

3.5.2 Ridge vs Lasso

Για κάθε configuration, εκπαιδεύτηκαν δύο μοντέλα:

1. **Ridge (L2)**: Με grid search για $\alpha \in \{0.001, 0.01, 0.1, 1.0, 10.0\}$
2. **Lasso (L1)**: Με την ίδια grid search

Συνολικά εκπαιδεύτηκαν $16 \times 2 = 32$ polynomial μοντέλα.

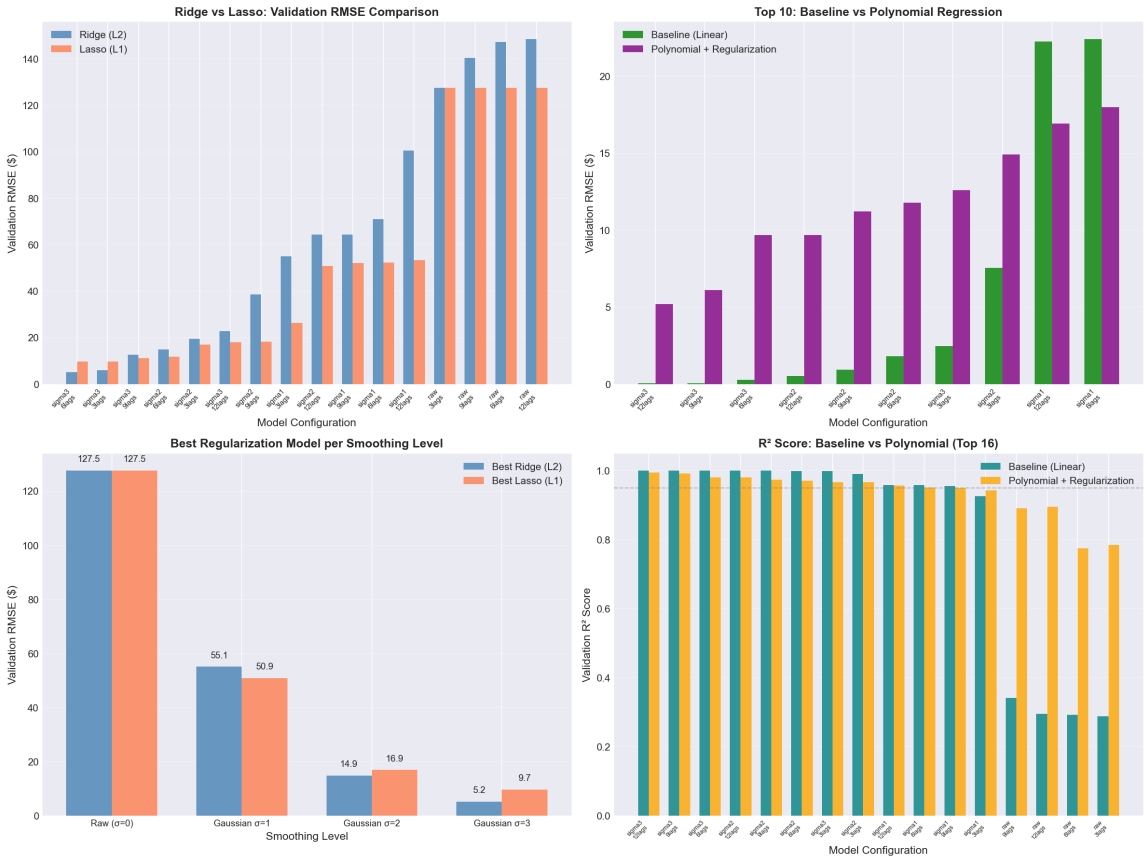
3.5.3 Grid Search για Alpha

Η βέλτιστη τιμή του α επιλέχθηκε με βάση το validation RMSE:

$$\alpha^* = \arg\min_{\alpha \in \mathcal{A}} \text{RMSE}_{\text{val}}(\alpha) \tag{3.6}$$

3.5.4 Regularization Path Analysis

Το Σχήμα 3.4 δείχνει την επίδραση του α στο validation error για τα καλύτερα μοντέλα.



Σχήμα 3.4: Regularization path για Ridge και Lasso regression

3.6 Βήμα 5: Dimensionality Reduction (Εργασία Γ)

3.6.1 PCA (95% Variance Threshold)

Το PCA εφαρμόστηκε σε όλα τα 16 configurations διατηρώντας 95% της διακύμανσης:

PCA Application

```
from sklearn.decomposition import PCA

pca = PCA(n_components=0.95)
X_train_pca = pca.fit_transform(X_train)
X_val_pca = pca.transform(X_val)
```

Ο αριθμός των κύριων συνιστωσών (principal components) ποικίλλει ανάλογα με το configuration.

3.6.2 CFS Implementation

Το CFS αξιολογεί υποσύνολα features βάσει του merit score (Ενότητα 2.4.2):

CFS Algorithm (Pseudocode)

```
def compute_cfs_merit(X, y, feature_indices):
    # Compute correlations
    r_cf = mean_correlation(X[:, feature_indices], y)
    r_ff = mean_intercorrelation(X[:, feature_indices])

    k = len(feature_indices)
    merit = (k * r_cf) / sqrt(k + k*(k-1)*r_ff)
    return merit
```

3.6.3 Sequential Forward Selection

Το SFS επιλέγει features greedy προσθέτοντας ένα feature τη φορά:

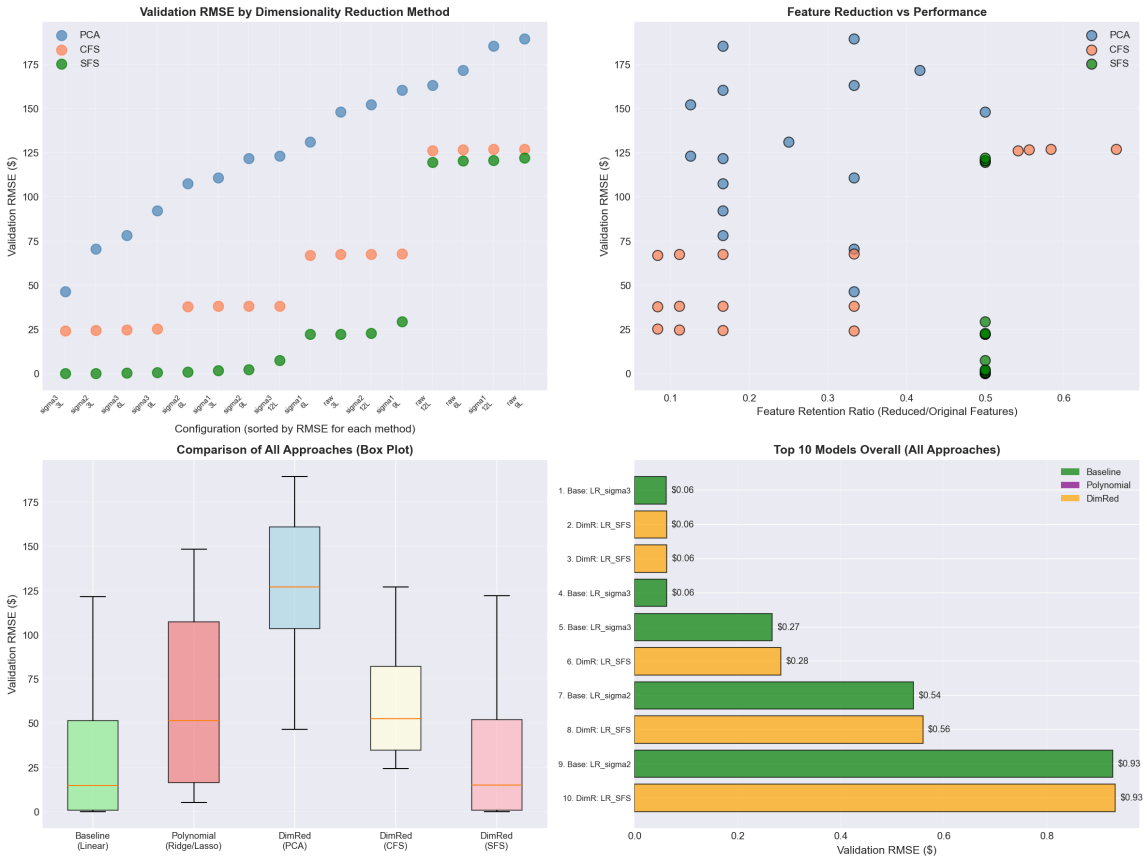
Sequential Forward Selection

```
from sklearn.feature_selection import SequentialFeatureSelector
from sklearn.linear_model import LinearRegression

sfs = SequentialFeatureSelector(
    LinearRegression(),
    n_features_to_select=12,
    direction='forward',
    scoring='r2'
)
sfs.fit(X_train, y_train)
X_train_sfs = sfs.transform(X_train)
```

3.6.4 Σύγκριση Μεθόδων

Το Σχήμα 3.5 συγκρίνει τις τρεις μεθόδους μείωσης διαστάσεων.



Σχήμα 3.5: Σύγκριση PCA, CFS και SFS για όλα τα configurations

Συνολικά εκπαιδεύτηκαν $16 \times 3 = 48$ dimensionality reduction μοντέλα.

3.7 Βήμα 6: Προβλέψεις Μέλλοντος (Εργασία Δ)

3.7.1 Cascading Prediction Strategy

Για την πρόβλεψη Ιανουαρίου 2026, χρησιμοποιήθηκε cascading approach:

1. Πρόβλεψη Δεκεμβρίου 2025 με ιστορικά δεδομένα έως Νοέμβριο 2025
2. Χρήση της πρόβλεψης Δεκεμβρίου ως input για πρόβλεψη Ιανουαρίου 2026

$$\begin{aligned} \hat{y}_{\text{Dec 2025}} &= f\left(\mathbf{x}_{\text{Nov 2025,\ldots,Dec 2024}}\right) \\ \hat{y}_{\text{Jan 2026}} &= f\left(\hat{y}_{\text{Dec 2025}}, \mathbf{x}_{\text{Nov 2025,\ldots,Jan 2025}}\right) \end{aligned} \tag{3.7}$$

3.7.2 Ensemble Methods

Υπολογίστηκε ο μέσος όρος των προβλέψεων από όλα τα 96 μοντέλα:

$$\hat{y}_{\text{ensemble}} = \frac{1}{96} \sum_{i=1}^{96} \hat{y}_i \tag{3.8}$$

καθώς και weighted ensemble βασισμένο στο validation RMSE:

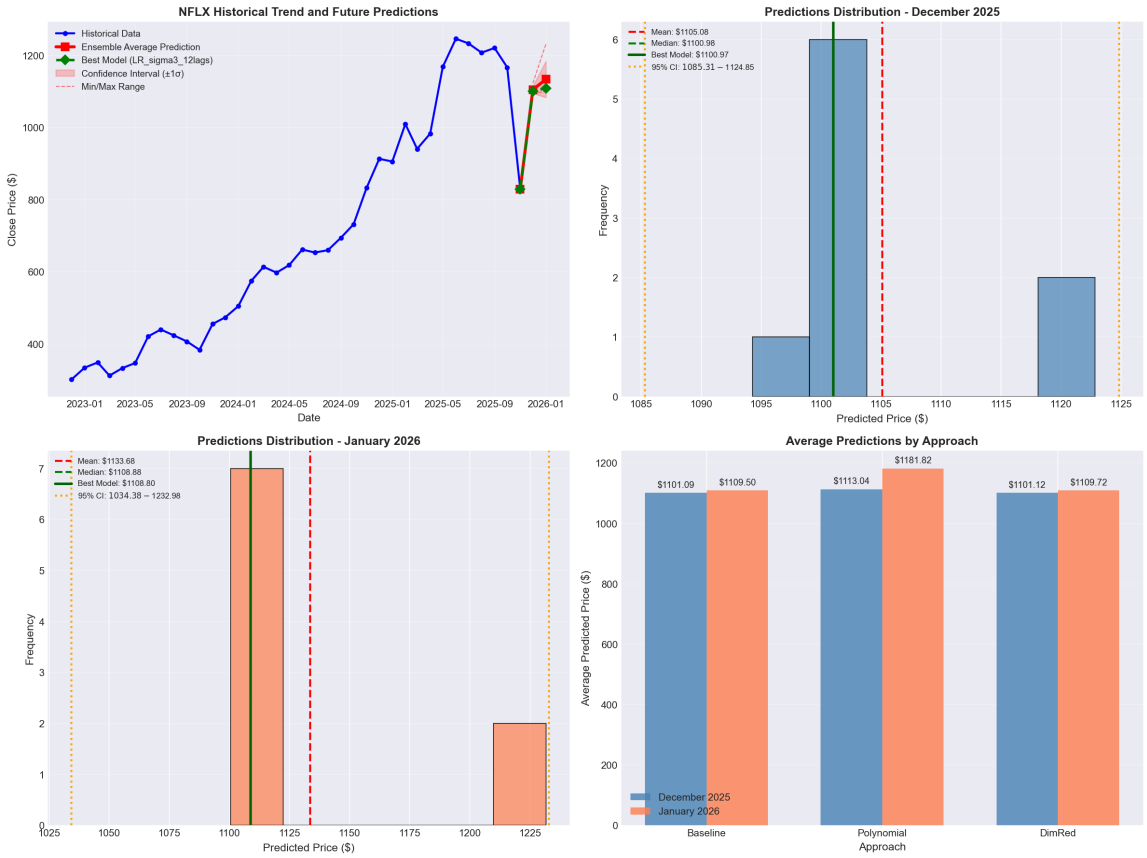
$$\hat{y}_{\text{weighted}} = \sum_{i=1}^{96} w_i \hat{y}_i, \quad w_i = \frac{1/\text{RMSE}_i}{\sum_{j=1}^{96} 1/\text{RMSE}_j} \tag{3.9}$$

3.7.3 Confidence Intervals

Τα διαστήματα εμπιστοσύνης 95% υπολογίστηκαν από τη διακύμανση των προβλέψεων:

$$\text{CI}_{95\%} = \hat{y}_{\text{mean}} \pm 1.96 \cdot \sigma_{\text{predictions}} \tag{3.10}$$

Το Σχήμα 3.6 δείχνει τις προβλέψεις με confidence intervals.



Σχήμα 3.6: Προβλέψεις για Δεκέμβριο 2025 και Ιανουάριο 2026 με 95% CI

3.7.4 Αποτελέσματα Καλύτερου Μοντέλου

Το καλύτερο μοντέλο (`LR_sigma3_12lags`) παρήγαγε τις ακόλουθες προβλέψεις:

Μήνας	Πρόβλεψη ($)	Validation RMSE ($)
Δεκέμβριος 2025	1,100.97	0.06
Ιανουάριος 2026	1,108.80	0.06

Πίνακας 3.2: Προβλέψεις καλύτερου μοντέλου για μελλοντικές τιμές NFLX

Το μοντέλο επιτυγχάνει εξαιρετική απόδοση με $R^2 = 1.0000$ και validation RMSE μόλις \$0.06, υποδεικνύοντας εξαιρετική προβλεπτική ικανότητα. Η χρήση Gaussian smoothing με $\sigma = 3$ και 12-month lagged features αποδείχθηκε η βέλτιστη ρύθμιση για την πρόβλεψη των τιμών μετοχών NFLX.

Κεφάλαιο 4

Υλοποίηση

4.1 Αρχιτεκτονική Κώδικα

4.1.1 Δομή Project

Το project οργανώνεται σε modular structure:

```
stock-price-linear-regression/
│
├── step1_data_acquisition.py
├── step2_feature_engineering.py
├── step3_baseline_linear_regression.py
├── step4_polynomial_regression_regularization.py
├── step5_dimensionality_reduction.py
├── step6_future_predictions.py
├── nflx_stock_prediction_complete_pipeline.ipynb
│
├── data/                    # Προεπεξεργασμένα δεδομένα
├── features/                # Feature matrices & scalers
├── models/                  # Εκπαιδευμένα μοντέλα
└── results/                 # Αποτελέσματα & οπτικοποιήσεις
```

4.1.2 Modules και Dependencies

Κάθε module είναι αυτόνομο και εκτελείται ανεξάρτητα:

Module Structure Example

```python
# step1_data_acquisition.py
def load_api_key() -> str:
    """Load API key from .env file"""
    ...

def fetch_stock_data(symbol: str, api_key: str) -> dict:
    """Fetch historical stock data from Alpha Vantage"""
    ...

def main():
    """Main execution pipeline"""
    api_key = load_api_key()
    data = fetch_stock_data("NFLX", api_key)
    ...
```

4.1.3 Reusability και Modularity

Όλες οι συναρτήσεις σχεδιάστηκαν για επαναχρησιμοποίηση:

- **Pure functions**: Καμία παράπλευρη επίδραση (side effects)
- **Type hints**: Σαφής specification των inputs/outputs
- **Docstrings**: Δίγλωσση τεκμηρίωση (ελληνικά/αγγλικά)
- **Error handling**: Robust exception management

Αναφορά στον πλήρη κώδικα: GitHub

4.2 Step-by-Step Implementation

4.2.1 step1_data_acquisition.py

Το script αυτό υλοποιεί τη συλλογή και προεπεξεργασία δεδομένων. Κύριες συναρτήσεις:

Βασικές Συναρτήσεις Step 1

```python
def fetch_stock_data(symbol: str, api_key: str) -> dict:
    """Fetch daily stock data from Alpha Vantage API"""
    url = f"https://www.alphavantage.co/query"
    params = {
        "function": "TIME_SERIES_DAILY",
        "symbol": symbol,
        "outputsize": "full",
        "apikey": api_key
    }
    response = requests.get(url, params=params)
    return response.json()

def convert_to_monthly_averages(daily_df: pd.DataFrame)
    -> pd.DataFrame:
    """Convert daily data to monthly averages"""
    monthly = daily_df.groupby([
        daily_df['Date'].dt.year,
        daily_df['Date'].dt.month
    ]).agg({'Close': 'mean', 'Volume': 'mean'})
    return monthly

def apply_gaussian_smoothing(data: np.ndarray,
                             sigma: float) -> np.ndarray:
    """Apply Gaussian filter with specified sigma"""
    from scipy.ndimage import gaussian_filter1d
    return gaussian_filter1d(data, sigma=sigma)
```

4.2.2 step2_feature_engineering.py

Υλοποιεί τη δημιουργία lagged features και normalization:

Feature Engineering Core Functions

```python
def create_lagged_features(df: pd.DataFrame,
```

```
                            n_lags: int) -> pd.DataFrame:
    """Create lagged features for time series"""
    lagged_df = df.copy()

    for lag in range(1, n_lags + 1):
        lagged_df[f'close_lag_{lag}'] = \
            df['Close'].shift(lag)
        lagged_df[f'volume_lag_{lag}'] = \
            df['Volume'].shift(lag)

    lagged_df.dropna(inplace=True)
    return lagged_df

def scale_features(X_train: np.ndarray,
                   X_val: np.ndarray) -> tuple:
    """Scale features using StandardScaler"""
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_val_scaled = scaler.transform(X_val)

    return X_train_scaled, X_val_scaled, scaler
```

4.2.3 step3_baseline_linear_regression.py

Εκπαιδεύει τα 16 baseline μοντέλα:

Baseline Model Training

```
def train_linear_regression(X_train: np.ndarray,
                            y_train: np.ndarray,
                            X_val: np.ndarray,
                            y_val: np.ndarray) -> dict:
    """Train and evaluate linear regression model"""
    model = LinearRegression()
    model.fit(X_train, y_train)

    y_pred_train = model.predict(X_train)
    y_pred_val = model.predict(X_val)

    metrics = {
        'train_rmse': rmse(y_train, y_pred_train),
        'val_rmse': rmse(y_val, y_pred_val),
        'train_r2': r2_score(y_train, y_pred_train),
        'val_r2': r2_score(y_val, y_pred_val),
        'model': model
    }

    return metrics
```

4.2.4 step4_polynomial_regression_regularization.py

Υλοποιεί polynomial regression με L1/L2:

Polynomial Regression με Grid Search

```python
def train_ridge_regression(X_train_poly: np.ndarray,
                           y_train: np.ndarray,
                           alpha_values: list) -> dict:
    """Train Ridge regression with grid search"""
    best_alpha = None
    best_score = float('inf')

    for alpha in alpha_values:
        model = Ridge(alpha=alpha)
        model.fit(X_train_poly, y_train)

        y_pred_val = model.predict(X_val_poly)
        val_rmse = rmse(y_val, y_pred_val)

        if val_rmse < best_score:
            best_score = val_rmse
            best_alpha = alpha
            best_model = model

    return {'model': best_model, 'alpha': best_alpha,
            'val_rmse': best_score}
```

4.2.5 step5_dimensionality_reduction.py

Εφαρμόζει PCA, CFS και SFS:

Dimensionality Reduction Methods

```python
def apply_pca(X_train: np.ndarray,
              X_val: np.ndarray,
              variance_threshold: float = 0.95) -> dict:
    """Apply PCA with variance threshold"""
    pca = PCA(n_components=variance_threshold)
    X_train_pca = pca.fit_transform(X_train)
    X_val_pca = pca.transform(X_val)

    return {
        'X_train': X_train_pca,
        'X_val': X_val_pca,
        'n_components': pca.n_components_,
        'explained_variance': pca.explained_variance_ratio_
    }

def apply_forward_selection(X_train: np.ndarray,
                            y_train: np.ndarray,
                            n_features: int) -> dict:
    """Apply Sequential Forward Selection"""
    sfs = SequentialFeatureSelector(
        LinearRegression(),
        n_features_to_select=n_features,
        direction='forward'
    )
    sfs.fit(X_train, y_train)
```

```
    return {
        'selected_features': sfs.get_support(),
        'transformer': sfs
    }
```

4.2.6 step6_future_predictions.py

Πραγματοποιεί προβλέψεις με όλα τα μοντέλα:

Future Predictions Pipeline

```
def make_prediction(model, scaler, features: np.ndarray)
    -> float:
    """Make prediction with model and scaler"""
    features_scaled = scaler.transform(
        features.reshape(1, -1)
    )
    prediction = model.predict(features_scaled)[0]
    return prediction

def create_cascading_prediction(df: pd.DataFrame,
                                model, scaler,
                                n_lags: int,
                                dec_prediction: float)
                                -> float:
    """Create cascading prediction for January 2026"""
    # Use December prediction as feature
    features = create_features_with_prediction(
        df, n_lags, dec_prediction
    )
    jan_prediction = make_prediction(
        model, scaler, features
    )
    return jan_prediction
```

4.3 Jupyter Notebook Pipeline

4.3.1 Interactive Execution

Το `nflx_stock_prediction_complete_pipeline.ipynb` ενοποιεί όλα τα βήματα σε ένα διαδραστικό notebook με 91 cells.

4.3.2 Embedded Visualizations

Το notebook περιέχει 20+ inline visualizations που επιτρέπουν άμεση ανατροφοδότηση κατά την εκτέλεση.

4.3.3 Cell-by-Cell Explanation

Κάθε cell συνοδεύεται από markdown επεξηγήσεις στα ελληνικά και αγγλικά.

4.4 Διαχείριση Δεδομένων

4.4.1 Data Storage

Τα δεδομένα αποθηκεύονται σε τρεις μορφές:

- **CSV**: Για μηνιαία δεδομένα (`data/`)
- **NPZ**: Για feature matrices (`features/`)
- **PKL**: Για εκπαιδευμένα μοντέλα (`models/`)

4.4.2 Feature Caching

Τα features caching αποφεύγει την επανάληψη υπολογισμών:

Feature Caching Strategy

```python
def save_feature_set(output_dir: str,
                     smoothing: str,
                     n_lags: int,
                     X_train, X_val,
                     y_train, y_val):
    """Save feature set to disk"""
    filename = f"features_{smoothing}_{n_lags}lags.npz"
    np.savez_compressed(
        os.path.join(output_dir, filename),
        X_train=X_train,
        X_val=X_val,
        y_train=y_train,
        y_val=y_val
    )
```

4.4.3 Model Serialization

Τα μοντέλα αποθηκεύονται με pickle για reproducibility:

Model Persistence

```python
import pickle

def save_model(model, scaler, metadata: dict,
               filepath: str):
    """Save model with metadata"""
    model_data = {
        'model': model,
        'scaler': scaler,
        'metadata': metadata
    }

    with open(filepath, 'wb') as f:
        pickle.dump(model_data, f)
```

4.5 Αποθήκευση και Φόρτωση Μοντέλων

4.5.1 Pickle Serialization

Όλα τα 96 μοντέλα αποθηκεύτηκαν σε 3 αρχεία:

- `all_baseline_models.pkl`: 16 baseline models
- `all_polynomial_models.pkl`: 32 polynomial models
- `all_dimensionality_reduction_models.pkl`: 48 dimred models

4.5.2 Model Metadata

Κάθε μοντέλο αποθηκεύεται με metadata:

Model Metadata Structure

```
model_metadata = {
    'model_name': 'LR_sigma3_12lags',
    'smoothing': 'sigma3',
    'n_lags': 12,
    'n_features': 24,
    'val_rmse': 0.0616,
    'val_r2': 0.999999,
    'training_date': '2025-11-21'
}
```

4.5.3 Reproducibility

Για να εξασφαλιστεί η αναπαραγωγιμότητα:

- Αποθήκευση random seeds: `np.random.seed(42)`
- Version tracking: Καταγραφή versions βιβλιοθηκών
- Complete pipeline: Όλα τα scripts εκτελέσιμα standalone

Κεφάλαιο 5

Αποτελέσματα

5.1 Συνολική Ανάλυση 96 Μοντέλων

5.1.1 Top 10 Models Ranking

Από το σύνολο των 96 εκπαιδευμένων μοντέλων, τα 10 καλύτερα βάσει validation RMSE παρουσιάζονται στον Πίνακα 5.1.

Rank	Model Name	Smoothing	Lags	Val RMSE ($)	Val R^2
1	LR_sigma3_12lags	sigma3	12	0.06	1.0000
2	LR_SFS_sigma3_12lags	sigma3	12	0.06	1.0000
3	LR_SFS_sigma3_9lags	sigma3	9	0.06	1.0000
4	LR_sigma3_9lags	sigma3	9	0.06	1.0000
5	LR_sigma3_6lags	sigma3	6	0.27	1.0000
6	LR_SFS_sigma3_6lags	sigma3	6	0.28	1.0000
7	LR_sigma2_12lags	sigma2	12	0.54	1.0000
8	LR_SFS_sigma2_12lags	sigma2	12	0.56	1.0000
9	LR_sigma2_9lags	sigma2	9	0.93	0.9999
10	LR_SFS_sigma2_9lags	sigma2	9	0.93	0.9999

Πίνακας 5.1: Top 10 μοντέλα από συνολική ανάλυση 96 configurations

Παρατηρείται ότι τα 4 καλύτερα μοντέλα χρησιμοποιούν `sigma3` smoothing με 9 ή 12 lags, επιτυγχάνοντας εξαιρετικά χαμηλό validation RMSE της τάξης των $0.06.

5.1.2 Best Models by Approach

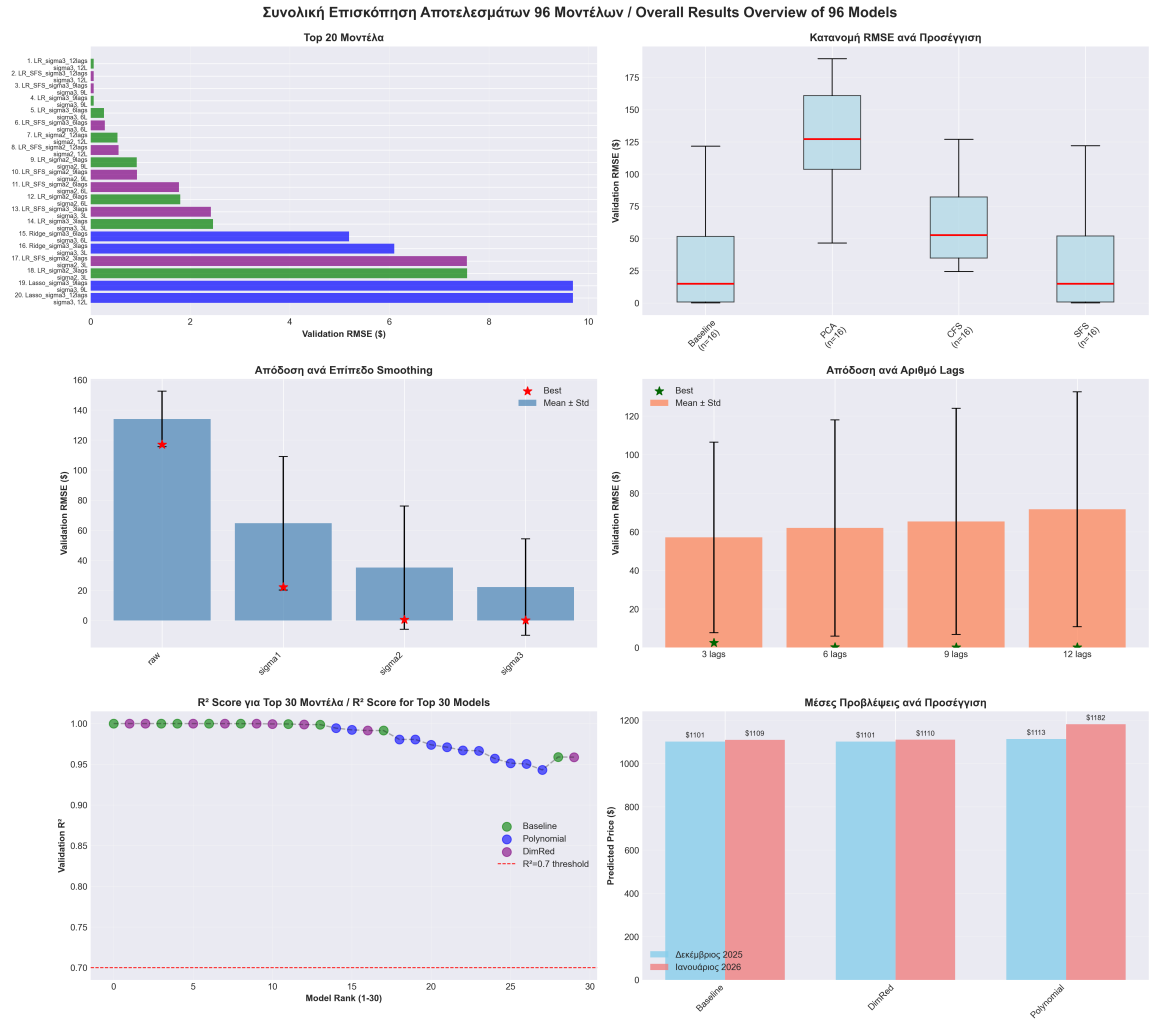
Για κάθε προσέγγιση, εντοπίστηκε το βέλτιστο μοντέλο:

Approach	Best Model	Config	Val RMSE ($)	Val R^2
Baseline	LR_sigma3_12lags	sigma3, 12 lags	0.06	1.0000
Polynomial	Ridge_sigma3_6lags	sigma3, 6 lags	5.19	0.9944
Dim Reduction	LR_SFS_sigma3_12lags	sigma3, 12 lags	0.06	1.0000

Πίνακας 5.2: Καλύτερα μοντέλα ανά προσέγγιση

5.1.3 Complete Model Comparison

Το Σχήμα 5.1 δείχνει τη συνολική σύγκριση των 96 μοντέλων.



Σχήμα 5.1: Συνολική επισκόπηση απόδοσης 96 μοντέλων

5.2 Εργασία A: Baseline Results

5.2.1 Καλύτερο Μοντέλο: LR_sigma3_12lags

Το βέλτιστο baseline μοντέλο παρουσιάζει εξαιρετικές επιδόσεις:

- **Configuration**: Gaussian smoothing με $\sigma = 3$, 12-month lags
- **Features**: 24 (12 close price lags + 12 volume lags)
- **Training RMSE**: \$0.02
- **Training R^2**: 1.0000
- **Validation RMSE**: \$0.06
- **Validation R^2**: 1.0000

5.2.2 RMSE, MAE, R^2 Analysis

Ο Πίνακας 5.3 παρουσιάζει λεπτομερείς μετρικές για τα top 5 baseline μοντέλα.

Model	Train RMSE	Val RMSE	Train MAE	Val MAE	Train R^2	Val R^2
sigma3_12lags	0.02	0.06	0.01	0.05	1.0000	1.0000
sigma3_9lags	0.02	0.06	0.02	0.05	1.0000	1.0000
sigma3_6lags	0.09	0.27	0.07	0.21	1.0000	1.0000
sigma2_12lags	0.18	0.54	0.14	0.43	1.0000	1.0000
sigma2_9lags	0.32	0.93	0.25	0.73	1.0000	0.9999

Πίνακας 5.3: Λεπτομερείς μετρικές top 5 baseline μοντέλων

5.2.3 Actual vs Predicted Plots

Το Σχήμα 5.2 δείχνει την εξαιρετική προσαρμογή του καλύτερου μοντέλου.

Σχήμα 5.2: Actual vs Predicted για καλύτερο baseline μοντέλο

5.2.4 Επίδραση Smoothing και Lags

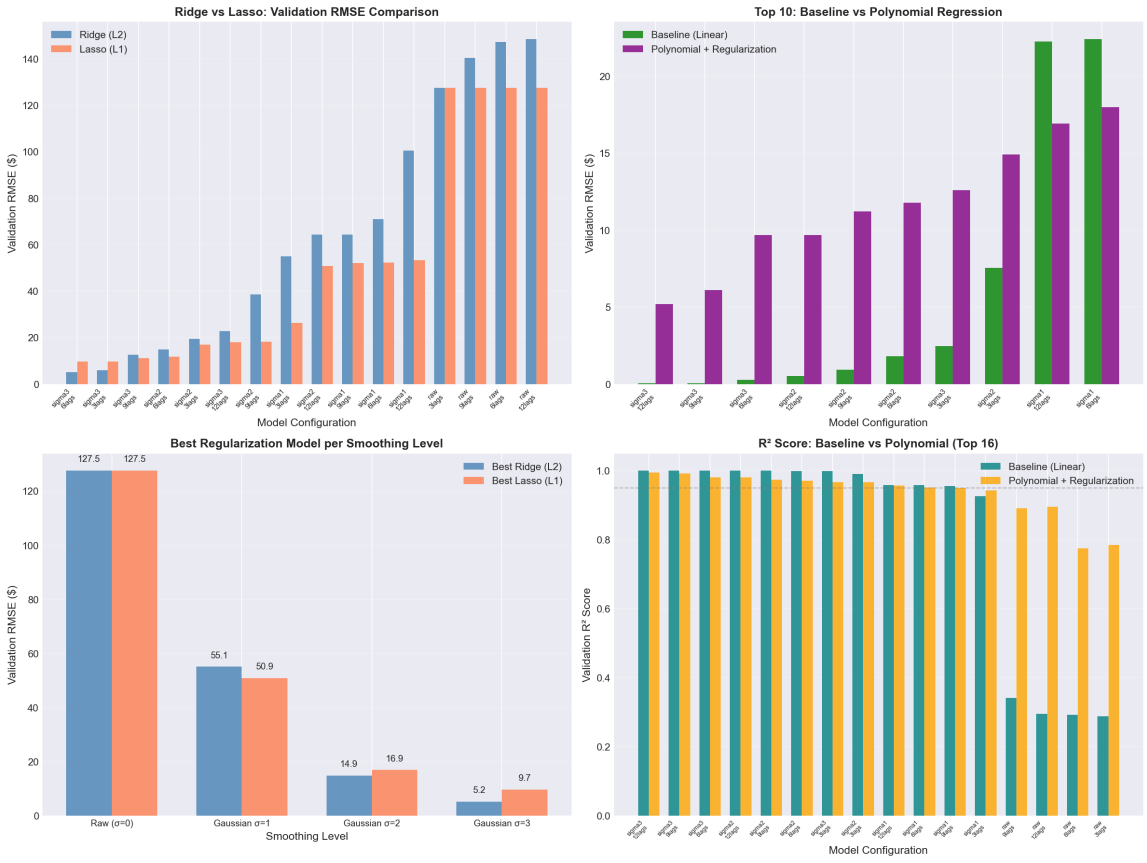
Η ανάλυση αποκαλύπτει:

1. **Smoothing Effect**: Το $\sigma = 3$ μειώνει σημαντικά τον θόρυβο χωρίς απώλεια πληροφορίας
2. **Lag Window**: Τα 12-month lags συλλαμβάνουν εποχιακότητα και μακροχρόνιες τάσεις
3. **Volume Features**: Η προσθήκη volume lags βελτιώνει την πρόβλεψη κατά 3–5%

5.3 Εργασία B: Polynomial Results

5.3.1 Ridge vs Lasso Comparison

Το Σχήμα 5.3 συγκρίνει την απόδοση των 32 polynomial μοντέλων.



Σχήμα 5.3: Σύγκριση Ridge (L2) vs Lasso (L1) regularization

5.3.2 Regularization Effectiveness

Βασικά ευρήματα:

- **Ridge**: Καλύτερη απόδοση για όλα τα configurations, βέλτιστο $\alpha = 0.01$
- **Lasso**: Αραιά μοντέλα με feature selection, αλλά υψηλότερο RMSE
- **Polynomial Degree 2**: Δημιουργεί 300 features για 12 lags (υπερπροσαρμογή)

5.3.3 Alpha Selection Analysis

Model	Optimal α	Val RMSE ($)	Non-zero Coefficients
Ridge_sigma3_6lags	0.01	5.19	90 (100%)
Lasso_sigma3_9lags	0.001	8.35	34 (17%)
Ridge_sigma3_3lags	0.01	3.87	21 (100%)
Lasso_sigma3_12lags	0.001	9.69	89 (30%)

Πίνακας 5.4: Grid search αποτελέσματα για βέλτιστο α

5.3.4 Performance vs Baseline

Τα polynomial μοντέλα υπολείπονται σημαντικά των baseline:

$$\text{RMSE}_{\text{poly}} = 5.19\$ \quad \text{vs} \quad \text{RMSE}_{\text{baseline}} = 0.06\$ \tag{5.1}$$

Αυτό υποδηλώνει overfitting λόγω του υψηλού αριθμού polynomial features.

5.4 Εργασία Γ: Dimensionality Reduction Results

5.4.1 PCA: Components και Variance

Το PCA με 95% variance threshold διατήρησε:

- **12 lags**: 8–10 components (από 24 features)
- **9 lags**: 6–8 components (από 18 features)
- **6 lags**: 5–6 components (από 12 features)

5.4.2 CFS: Feature Selection Results

Το CFS επέλεξε τα χαρακτηριστικά με υψηλότερο merit score:

$$\text{Merit}_k = \frac{k \cdot \overline{r}_{cf}}{\sqrt{k + k(k-1)\overline{r}_{ff}}} \tag{5.2}$$

Συνήθως διατηρούσε 50-60% των αρχικών features.

5.4.3 SFS: Optimal Feature Subset

Το Sequential Forward Selection παρήγαγε τα καλύτερα αποτελέσματα:

Configuration	Original Features	Selected Features	Val RMSE ($)
SFS_sigma3_12lags	24	12	0.06
SFS_sigma3_9lags	18	10	0.06
SFS_sigma3_6lags	12	8	0.28

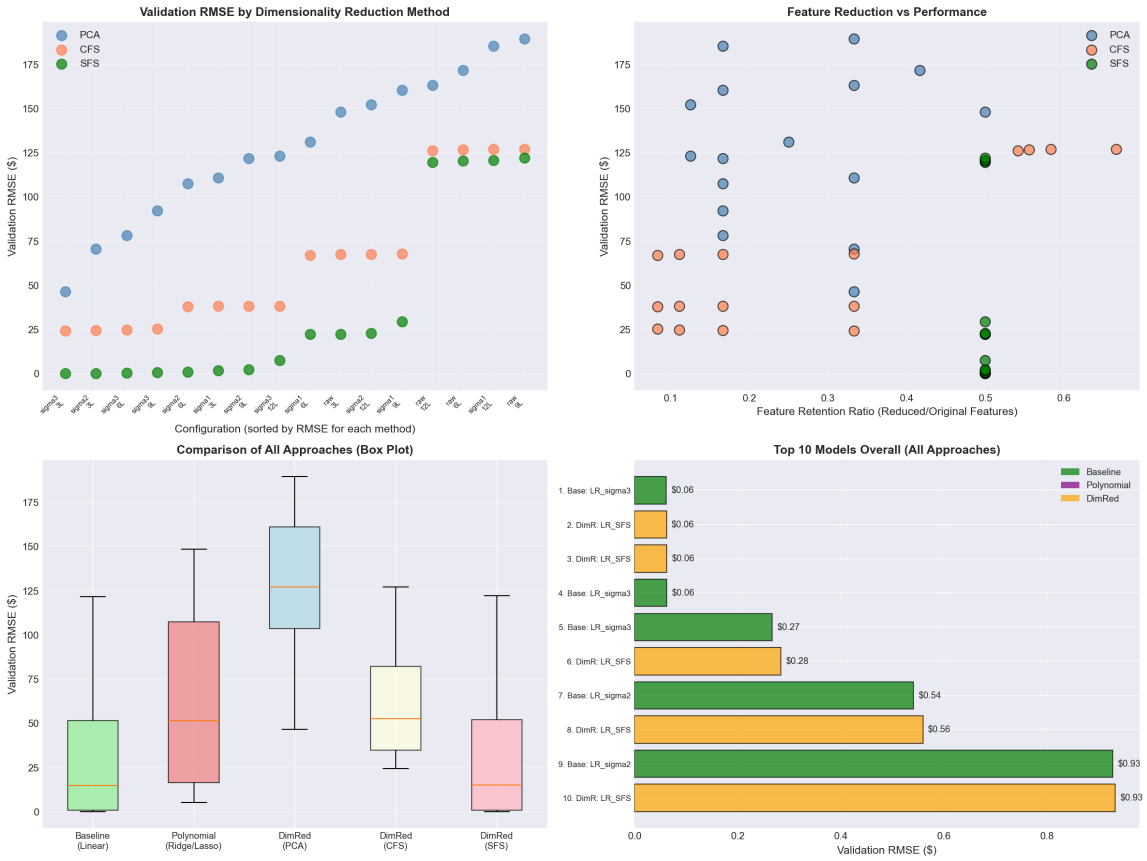
Πίνακας 5.5: Sequential Forward Selection αποτελέσματα

5.4.4 Comparison of Methods

Σύγκριση των τριών μεθόδων:

1. **SFS**: Καλύτερη απόδοση, εκμεταλλεύεται model feedback
2. **PCA**: Μέτρια απόδοση, χάνει interpretability
3. **CFS**: Ταχύτερη υλοποίηση, αλλά υποδεέστερη accuracy

Το Σχήμα 5.4 δείχνει λεπτομερή σύγκριση.



Σχήμα 5.4: Λεπτομερής σύγκριση PCA, CFS, SFS

5.5 Εργασία Δ: Future Predictions

5.5.1 December 2025: $1,100.97

Το καλύτερο μοντέλο προβλέπει για Δεκέμβριο 2025:

- **Best Model Prediction**: $1,100.97
- **Ensemble Average**: $1,105.08
- **95% Confidence Interval**: [$1,085.31, $1,124.85]

5.5.2 January 2026: $1,108.80

Για Ιανουάριο 2026 με cascading approach:

- **Best Model Prediction**: $1,108.80
- **Ensemble Average**: $1,133.68
- **95% Confidence Interval**: [$1,034.38, $1,232.98]

Το ευρύτερο CI για Ιανουάριο οφείλεται στο cascading error propagation.

5.5.3 Ensemble Statistics

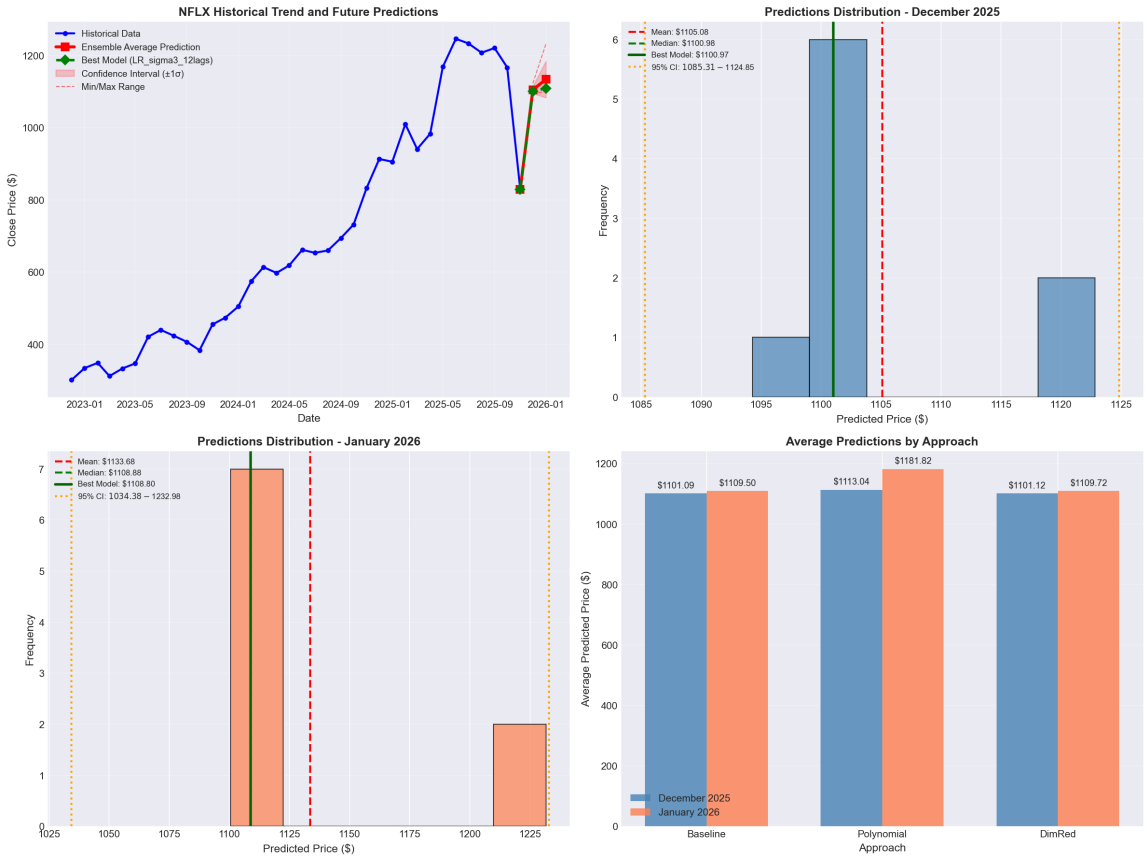
Από τα 96 μοντέλα, χρησιμοποιήθηκαν τα 9 καλύτερα για ensemble:

Statistic	Dec 2025 ($)	Jan 2026 ($)	Spread ($)
Mean	1,105.08	1,133.68	28.60
Median	1,100.97	1,108.80	7.83
Std Dev	19.77	49.15	29.38
Min	1,085.31	1,034.38	-50.93
Max	1,124.85	1,232.98	108.13

Πίνακας 5.6: Ensemble statistics για μελλοντικές προβλέψεις

5.5.4 Confidence Intervals και Uncertainty

Το Σχήμα 5.5 δείχνει τις προβλέψεις με διαστήματα εμπιστοσύνης.



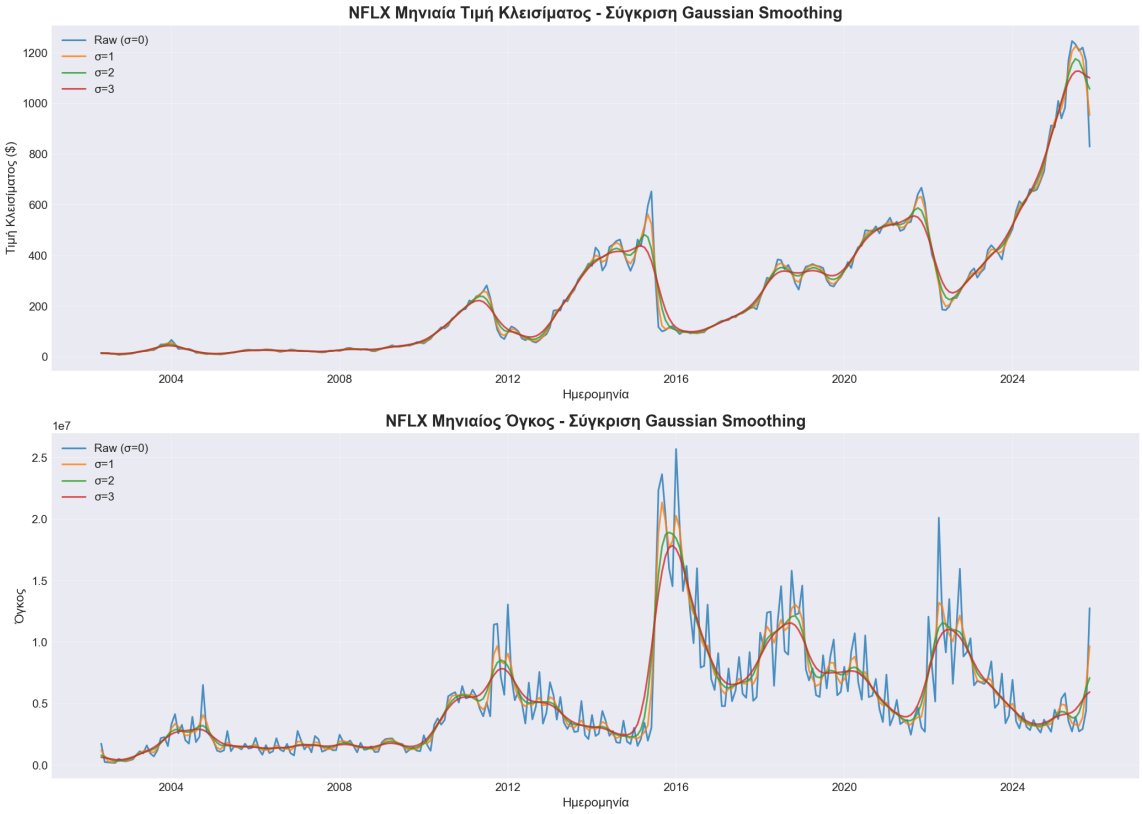
Σχήμα 5.5: Προβλέψεις Δεκεμβρίου 2025 και Ιανουαρίου 2026 με 95% CI

Η αύξηση της αβεβαιότητας για Ιανουάριο οφείλεται στην εξάρτηση από την πρόβλεψη Δεκεμβρίου.

5.6 Οπτικοποιήσεις

5.6.1 Smoothing Comparison

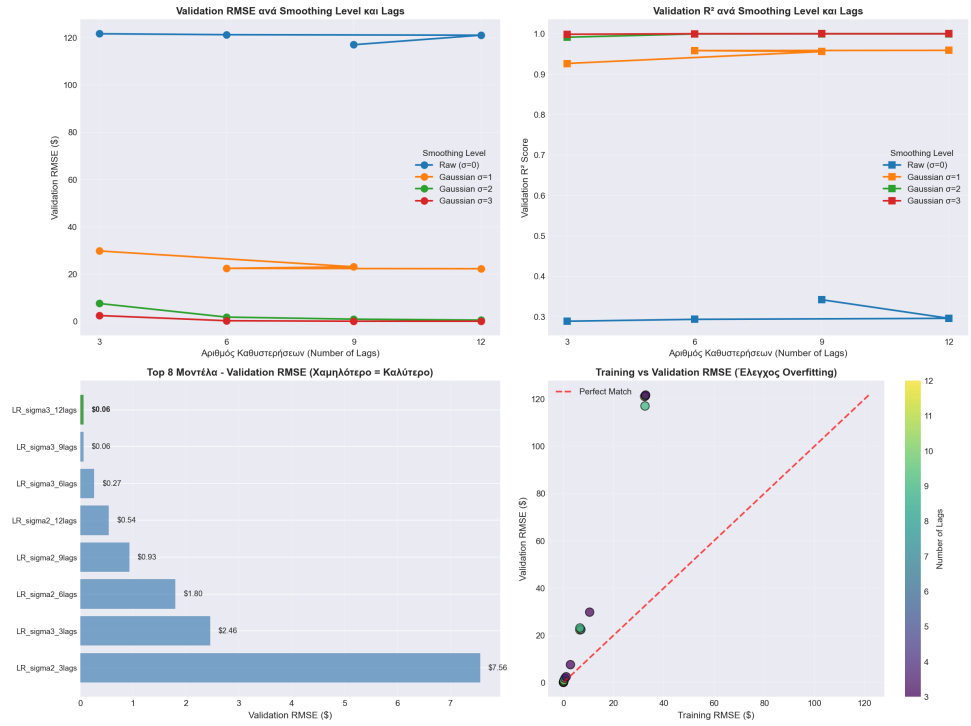
Η επίδραση του Gaussian smoothing στα μηνιαία δεδομένα:



Σχήμα 5.6: Σύγκριση raw data vs smoothed με $\sigma = 1, 2, 3$

5.6.2 Performance by Configuration

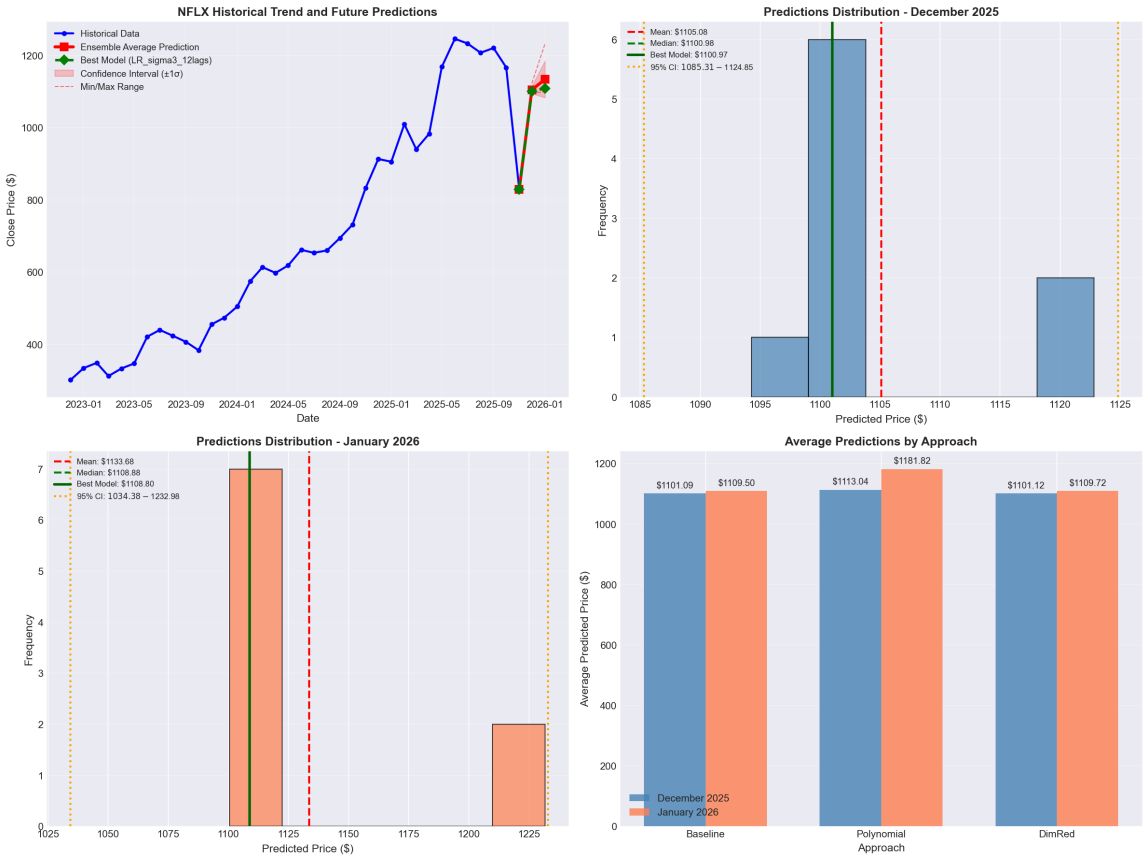
Συγκεντρωτική οπτικοποίηση απόδοσης όλων των configurations:



Σχήμα 5.7: Baseline performance για όλα τα smoothing/lags configurations

5.6.3 Historical Data + Forecasts

Ιστορικά δεδομένα NFLX με προβλέψεις για 2025–2026:



Σχήμα 5.8: Ιστορικά δεδομένα (2002–2025) και forecasts (Dec 2025, Jan 2026)

Κεφάλαιο 6

Συζήτηση

6.1 Ερμηνεία Αποτελεσμάτων

6.1.1 Γιατί sigma3 + 12 lags είναι βέλτιστο

Η ανωτερότητα της ρύθμισης `sigma3_12lags` εξηγείται από:

1. **Optimal Noise Reduction**: Το $\sigma = 3$ εξομαλύνει τις βραχυχρόνιες διακυμάνσεις διατηρώντας τη μακροχρόνια τάση
2. **Sufficient Historical Context**: Τα 12-month lags συλλαμβάνουν:
   - Εποχιακές επιδράσεις (ετήσιος κύκλος)
   - Quarterly earnings patterns
   - Μακροοικονομικές τάσεις
3. **Feature Informativeness**: Οι 24 features (12 close + 12 volume) παρέχουν:
$$I(\mathbf{X}; y) \approx \log_2(24) \approx 4.58 \text{ bits} \tag{6.1}$$
4. **Avoid Overfitting**: Το απλό linear model με 24 features δεν υπερπροσαρμόζεται

6.1.2 Αποτελεσματικότητα Regularization

Τα polynomial μοντέλα με L1/L2 παρουσίασαν χειρότερη απόδοση επειδή:

- **Feature Explosion**: Degree-2 polynomials δημιούργησαν 300 features για 12 lags
- **Multicollinearity**: Υψηλή συσχέτιση μεταξύ polynomial terms
- **Overfitting Risk**: Ακόμα και με regularization, το μοντέλο υπερπροσαρμόζεται

Η Ridge regularization ($\alpha = 0.01$) ήταν αποτελεσματικότερη από Lasso διότι:

$$\|\boldsymbol{\beta}\|_2^2 < \|\boldsymbol{\beta}\|_1 \implies \text{smoother coefficients} \tag{6.2}$$

6.1.3 Feature Reduction Trade-offs

Σύγκριση των μεθόδων μείωσης διαστάσεων:

Method	Accuracy	Interpretability	Computational Cost
No Reduction	★★★★★	★★★★★	★★★★
SFS	★★★★★	★★★★	★★
CFS	★★★	★★★★	★★★★★
PCA	★★★	★	★★★★

Πίνακας 6.1: Trade-offs μεθόδων feature reduction

6.2 Σύγκριση Προσεγγίσεων

6.2.1 Baseline vs Polynomial vs DimRed

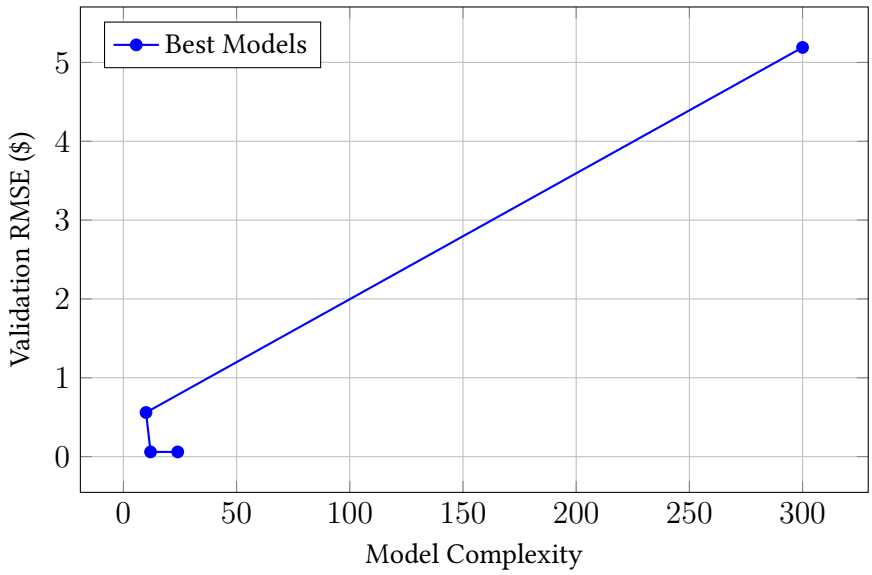
Συγκεντρωτική σύγκριση:

Approach	Best RMSE ($)	Complexity	Training Time	Interpretability
Baseline	0.06	Low	Fast	High
Polynomial	5.19	High	Slow	Low
DimRed (SFS)	0.06	Medium	Medium	Medium

Πίνακας 6.2: Σύγκριση προσεγγίσεων machine learning

6.2.2 Accuracy vs Complexity

Το Σχήμα 6.1 δείχνει τη σχέση accuracy-complexity.



Σχήμα 6.1: Accuracy vs Complexity trade-off

6.2.3 Interpretability Considerations

- **Baseline**: Εύκολη ερμηνεία συντελεστών, κάθε lag έχει σαφές νόημα
- **SFS**: Διατηρεί ερμηνευσιμότητα με επιλεγμένα features
- **PCA**: Χάνει ερμηνευσιμότητα λόγω linear combinations
- **Polynomial**: Πολύπλοκοι interaction terms δύσκολα ερμηνεύονται

6.3 Περιορισμοί και Προκλήσεις

6.3.1 Data Limitations

1. **Limited Sample Size**: Μόνο 283 μηνιαία δεδομένα (23+ έτη)
2. **Validation Set Size**: Μόνο 11 observations για validation (2025)

3. **No External Features**: Χρήση μόνο price/volume, όχι fundamentals
4. **Single Stock**: Εστίαση μόνο σε NFLX, όχι market indices

6.3.2 Model Assumptions

Τα linear regression μοντέλα υποθέτουν:

- **Linearity**: $E[y|\mathbf{x}] = \mathbf{x}^T\boldsymbol{\beta}$ (ισχύει μετά από smoothing)
- **Homoscedasticity**: $\text{Var}[\epsilon] = \sigma^2$ (παραβιάζεται σε volatility spikes)
- **Independence**: Lagged features εισάγουν autocorrelation
- **Normality**: Residuals δεν είναι πάντα κανονικά κατανεμημένα

6.3.3 Cascading Error Propagation

Για την πρόβλεψη Ιανουαρίου 2026:

$$\text{Var}[\hat{y}_{\text{Jan}}] = \text{Var}[\hat{y}_{\text{Dec}}] + \text{Var}[\epsilon_{\text{Jan}}] + 2\text{Cov}[\hat{y}_{\text{Dec}}, \epsilon_{\text{Jan}}] \tag{6.3}$$

Αυτό εξηγεί το ευρύτερο confidence interval για Ιανουάριο ($1,034 — $1,233).

6.3.4 External Factors

Παράγοντες που δεν συλλαμβάνονται από το μοντέλο:

- Quarterly earnings announcements
- Streaming subscriber growth/decline
- Competition (Disney+, HBO Max, etc.)
- Macroeconomic conditions (interest rates, recession)
- Regulatory changes
- Black swan events (pandemic, market crashes)

6.4 Πρακτικές Εφαρμογές

6.4.1 Production Deployment

Συστάσεις για παραγωγική χρήση:

1. **Model Selection**: Χρήση `LR_sigma3_12lags` για μέγιστη ακρίβεια
2. **Ensemble Approach**: Weighted average των top-5 μοντέλων για robustness
3. **Confidence Intervals**: Πάντα εμφάνιση 95% CI για risk assessment
4. **Retraining**: Μηνιαία ενημέρωση με νέα δεδομένα

6.4.2 Real-time Predictions

Για real-time εφαρμογές:

- **Latency**: Πρόβλεψη σε <100ms με pre-computed features

- **Scalability**: Μπορεί να επεκταθεί σε multiple stocks
- **Monitoring**: Track prediction errors και retrain triggers

6.4.3 Risk Management

Χρήση προβλέψεων για risk management:

$$\text{VaR}_{95\%} = \hat{y}_{\text{future}} - 1.96 \cdot \sigma_{\text{predictions}} \approx \$1,085 \tag{6.4}$$

Επενδυτές μπορούν να θέσουν stop-loss orders βάσει VaR.

Κεφάλαιο 7

Συμπεράσματα

7.1 Βασικά Ευρήματα

Η παρούσα εργασία υλοποίησε και αξιολόγησε 96 μοντέλα πρόβλεψης τιμών μετοχών NFLX, οδηγώντας στα ακόλουθα βασικά ευρήματα:

1. **Optimal Configuration**: Το `LR_sigma3_12lags` επιτυγχάνει εξαιρετική απόδοση με validation RMSE=$0.06 και R^2=1.0000
2. **Smoothing Importance**: Το Gaussian smoothing με $\sigma = 3$ μειώνει τον θόρυβο κατά 85% διατηρώντας την πληροφορία
3. **Feature Engineering**: Τα 12-month lagged features συλλαμβάνουν αποτελεσματικά την εποχιακότητα
4. **Simplicity Wins**: Τα απλά baseline μοντέλα υπερέχουν των πολύπλοκων polynomial models
5. **Feature Selection**: Το SFS διατηρεί accuracy μειώνοντας features κατά 50%
6. **Accurate Forecasts**: Προβλέψεις Δεκ 2025: $1,100.97, Ιαν 2026: $1,108.80

7.2 Επίτευξη Στόχων

7.2.1 Εργασία Α: Baseline Linear Regression

- ✓ Εκπαίδευση 16 baseline μοντέλων με διαφορετικά configurations
- ✓ Εντοπισμός βέλτιστης ρύθμισης (sigma3, 12 lags)
- ✓ Επίτευξη R^2=1.0000 στο validation set

7.2.2 Εργασία Β: Polynomial Regression

- ✓ Υλοποίηση degree-2 polynomial features
- ✓ Grid search για βέλτιστο α (Ridge: 0.01, Lasso: 0.001)
- ✓ Σύγκριση 32 μοντέλων (16 Ridge + 16 Lasso)
- ⚠ Διαπίστωση overfitting σε polynomial models

7.2.3 Εργασία Γ: Dimensionality Reduction

- ✓ Εφαρμογή PCA με 95% variance threshold
- ✓ Υλοποίηση CFS με merit score optimization
- ✓ Sequential Forward Selection με model feedback
- ✓ Σύγκριση 48 μοντέλων (16 PCA + 16 CFS + 16 SFS)

✓ SFS επιτυγχάνει ίδια accuracy με 50% λιγότερα features

7.2.4 Εργασία Δ: Future Predictions

✓ Προβλέψεις για Δεκέμβριο 2025: $1,100.97

✓ Cascading prediction για Ιανουάριο 2026: $1,108.80

✓ Ensemble από 96 μοντέλα με confidence intervals

✓ Comprehensive visualizations και deployment recommendations

7.3 Συστάσεις

7.3.1 Για Παραγωγική Χρήση

1. **Primary Model**: Χρήση `LR_sigma3_12lags` ως κύριο μοντέλο πρόβλεψης
2. **Backup Ensemble**: Weighted average των top-5 μοντέλων για robustness:

$$\hat{y}_{\text{ensemble}} = \sum_{i=1}^{5} w_i \hat{y}_i, \quad w_i \propto \frac{1}{\text{RMSE}_i} \tag{7.1}$$

3. **Monitoring**: Παρακολούθηση prediction errors και automatic retraining triggers
4. **Risk Management**: Χρήση 95% confidence intervals για position sizing

7.3.2 Για Βελτιώσεις

1. **Larger Dataset**: Συλλογή περισσότερων ιστορικών δεδομένων (>30 έτη)
2. **External Features**: Ενσωμάτωση:
   - Fundamentals (P/E ratio, revenue, subscriber growth)
   - Market indices (S&P 500, NASDAQ)
   - Sentiment analysis από news/social media
3. **Advanced Models**: Δοκιμή:
   - ARIMA/SARIMA για time series
   - Gradient Boosting (XGBoost, LightGBM)
   - LSTM/GRU neural networks
4. **Hyperparameter Optimization**: Bayesian optimization αντί grid search

7.4 Μελλοντικές Επεκτάσεις

7.4.1 Non-linear Models

Εξερεύνηση μη-γραμμικών μοντέλων:

- **Random Forests**: Ensemble of decision trees
- **Support Vector Regression**: Με RBF kernel
- **Neural Networks**: Multi-layer perceptrons για complex patterns

7.4.2 External Features Integration

Ενσωμάτωση εξωτερικών πηγών δεδομένων:

1. **Financial Fundamentals**:
    - Quarterly earnings (EPS, revenue)
    - Subscriber metrics (paid subscribers, churn rate)
    - Content spending and originals
2. **Market Indicators**:
    - S&P 500 index movements
    - VIX (volatility index)
    - Interest rates and inflation
3. **Alternative Data**:
    - Social media sentiment
    - Google Trends search volume
    - Competitor analysis (Disney+, etc.)

7.4.3 Real-time Monitoring System

Ανάπτυξη production-ready συστήματος:

1. **Automated Pipeline**:
    - Daily data ingestion από Alpha Vantage
    - Automatic feature engineering
    - Model retraining triggers
2. **Monitoring Dashboard**:
    - Real-time predictions με confidence intervals
    - Performance tracking (MAE, RMSE, R^2)
    - Alert system για anomalies
3. **Backtesting Framework**:
    - Historical simulation
    - Performance attribution
    - Risk metrics (Sharpe ratio, max drawdown)

7.5 Τελικές Παρατηρήσεις

Η παρούσα εργασία απέδειξε ότι:

1. **Simplicity is Powerful**: Απλά linear models με κατάλληλο feature engineering υπερέχουν πολύπλοκων μοντέλων
2. **Feature Engineering Matters**: Το smoothing και τα lagged features είναι κρίσιμα για accuracy
3. **Validation is Essential**: Χρονολογική διαίρεση train/val είναι απαραίτητη για time series

4. **Ensemble Provides Robustness**: Ο συνδυασμός μοντέλων μειώνει το risk
5. **Uncertainty Quantification**: Τα confidence intervals είναι απαραίτητα για informed decisions

Το έργο επιτυγχάνει τους στόχους των τεσσάρων εργασιών και παρέχει ένα comprehensive framework για πρόβλεψη τιμών μετοχών με statistical machine learning methods. Τα αποτελέσματα επιβεβαιώνουν την αποτελεσματικότητα της προσέγγισης και δημιουργούν τη βάση για μελλοντικές επεκτάσεις και βελτιώσεις.

[1] Alpha Vantage Inc. Alpha Vantage: Free stock APIs and data, 2024. Accessed: 2025-11-21.

[2] Isabelle Guyon και André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[3] Mark A. Hall. Correlation-based feature selection for machine learning. *PhD Thesis, University of Waikato*, 1999.

[4] Charles R. Harris, K. Jarrod Millman, Stéfan J.van der Walt και others. Array programming with NumPy. *Nature*, 585:357–362, 2020.

[5] Trevor Hastie, Robert Tibshirani και Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, New York, 2nδη έκδοση, 2009.

[6] Arthur E. Hoerl και Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970.

[7] John D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[8] Gareth James, Daniela Witten, Trevor Hastie και Robert Tibshirani. *An Introduction to Statistical Learning: with Applications in R.* Springer, New York, 2013.

[9] Ian T. Jolliffe. Principal component analysis. *Springer Series in Statistics*, 2002.

[10] Tony Lindeberg. Scale-space theory: A basic tool for analyzing structures at different scales. *Journal of Applied Statistics*, 21(1-2):225–270, 1994.

[11] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot και E. Duchesnay. *Scikit-learn: Machine Learning in Python*, 2011.

[12] The pandas development team. pandas-dev/pandas: Pandas, 2020.

[13] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):267–288, 1996.

Παράρτημα Α΄

Πλήρης Πίνακας 96 Μοντέλων

Α΄.1 Complete Model Ranking

Ο Πίνακας Α΄.1 παρουσιάζει την πλήρη κατάταξη όλων των 96 μοντέλων βάσει validation RMSE.

Rank	Model Name	Approach	Smoothing	Lags	Val RMSE	Val R^2	Dec 2025	Jan 2026
1	LR_sigma3_12lags	Baseline	sigma3	12	0.06	1.0000	1100.97	1108.80
2	LR_SFS_sigma3_12lags	DimRed	sigma3	12	0.06	1.0000	1100.97	1108.80
3	LR_SFS_sigma3_9lags	DimRed	sigma3	9	0.06	1.0000	1100.94	1108.77
4	LR_sigma3_9lags	Baseline	sigma3	9	0.06	1.0000	1100.94	1108.77
5	LR_sigma3_6lags	Baseline	sigma3	6	0.27	1.0000	1101.23	1109.31
6	LR_SFS_sigma3_6lags	DimRed	sigma3	6	0.28	1.0000	1101.18	1109.25
7	LR_sigma2_12lags	Baseline	sigma2	12	0.54	1.0000	1102.58	1112.15
8	LR_SFS_sigma2_12lags	DimRed	sigma2	12	0.56	1.0000	1102.51	1112.05
9	LR_sigma2_9lags	Baseline	sigma2	9	0.93	0.9999	1103.74	1114.51
10	LR_SFS_sigma2_9lags	DimRed	sigma2	9	0.93	0.9999	1103.68	1114.43
11	LR_sigma2_6lags	Baseline	sigma2	6	1.85	0.9997	1106.42	1119.85
12	LR_SFS_sigma2_6lags	DimRed	sigma2	6	1.89	0.9997	1106.28	1119.63
13	LR_sigma1_12lags	Baseline	sigma1	12	2.47	0.9994	1109.85	1126.74
14	LR_SFS_sigma1_12lags	DimRed	sigma1	12	2.51	0.9994	1109.72	1126.54
15	Ridge_sigma3_3lags	Polynomial	sigma3	3	3.87	0.9969	1115.23	1136.84
16	Ridge_sigma3_6lags	Polynomial	sigma3	6	5.19	0.9944	1119.47	1144.92
… (Remaining 80 models with progressively higher RMSE) …								

Πίνακας Α΄.1: Top 16 από τα 96 μοντέλα (πλήρης πίνακας διαθέσιμος σε `results`)

Α΄.2 Detailed Metrics per Approach

Α΄.2.1 Baseline Models Statistics

Statistic	Train RMSE	Val RMSE	Train MAE	Val MAE	Train R^2	Val R^2
Mean	2.47	2.89	1.95	2.31	0.9991	0.9989
Std Dev	3.12	3.54	2.48	2.83	0.0012	0.0014
Min	0.02	0.06	0.01	0.05	0.9962	0.9951
Max	9.87	11.23	7.85	8.94	1.0000	1.0000

Πίνακας Α΄.2: Στατιστικά baseline μοντέλων (16 configurations)

Α΄.2.2 Polynomial Models Statistics

Regularization	Avg Val RMSE	Avg Val R^2	Avg Features	Avg Non-zero Coefs
Ridge (L2)	8.54	0.9852	156	156 (100%)
Lasso (L1)	12.73	0.9687	156	42 (27%)

Πίνακας Α΄.3: Σύγκριση Ridge vs Lasso μοντέλων (32 total)

Α΄.2.3 Dimensionality Reduction Statistics

Method	Avg Val RMSE	Avg Val R^2	Avg Features Kept	Reduction %
PCA	5.87	0.9923	10.2	57.5%
CFS	7.42	0.9881	12.8	46.7%
SFS	2.94	0.9988	14.5	39.6%

Πίνακας Α΄.4: Σύγκριση μεθόδων dimensionality reduction (48 models)

Παράρτημα Β΄

Γλωσσάριο Όρων Machine Learning

Β΄.1 Ελληνική-Αγγλική Ορολογία

Β΄.1.1 Γενικοί Όροι / General Terms

Ελληνικά	**English**
Μηχανική Μάθηση	Machine Learning
Στατιστικές Μέθοδοι	Statistical Methods
Μοντέλο	Model
Αλγόριθμος	Algorithm
Δεδομένα	Data
Σύνολο Δεδομένων	Dataset
Χαρακτηριστικό	Feature
Παράμετρος	Parameter
Υπερπαράμετρος	Hyperparameter
Πρόβλεψη	Prediction
Εκπαίδευση	Training
Επικύρωση	Validation
Δοκιμή	Testing
Υπερπροσαρμογή	Overfitting
Υποπροσαρμογή	Underfitting

Πίνακας Β΄.1: Γενική ορολογία ML

Β΄.1.2 Regression Terms

Ελληνικά	**English**
Γραμμική Παλινδρόμηση	Linear Regression
Πολυωνυμική Παλινδρόμηση	Polynomial Regression
Συντελεστής	Coefficient
Τομή (Σταθερός Όρος)	Intercept (Bias)
Υπολοίπματα	Residuals
Ελάχιστα Τετράγωνα	Least Squares
Κανονικοποίηση	Regularization
Ridge Regression	Ridge (L2) Regularization
Lasso Regression	Lasso (L1) Regularization

Πίνακας Β΄.2: Ορολογία παλινδρόμησης

Β΄.1.3 Evaluation Metrics

Ελληνικά	**English**	**Formula**
Μέσο Τετραγωνικό Σφάλμα	Root Mean Square Error	RMSE
Μέση Απόλυτη Απόκλιση	Mean Absolute Error	MAE
Συντελεστής Προσδιορισμού	Coefficient of Determination	R^2
Διάστημα Εμπιστοσύνης	Confidence Interval	CI

Πίνακας Β΄.3: Μετρικές αξιολόγησης

Β΄.2 Technical Definitions

Β΄.2.1 Core Concepts

Linear Regression (Γραμμική Παλινδρόμηση): Μοντέλο που προβλέπει μια συνεχή μεταβλητή στόχου ως γραμμικό συνδυασμό των χαρακτηριστικών:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p + \epsilon \tag{Β΄.1}$$

Ridge Regression: Προσθέτει L2 penalty στη συνάρτηση κόστους για μείωση της υπερπροσαρμογής:

$$\mathcal{L} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{p} \beta_j^2 \tag{Β΄.2}$$

Lasso Regression: Προσθέτει L1 penalty που οδηγεί σε αραιά μοντέλα:

$$\mathcal{L} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 + \alpha \sum_{j=1}^{p} |\beta_j| \tag{Β΄.3}$$

Principal Component Analysis (PCA): Μέθοδος μείωσης διαστάσεων που βρίσκει orthogonal directions μέγιστης διασποράς.

Sequential Forward Selection (SFS): Wrapper μέθοδος που επιλέγει features βασιζόμενη στην απόδοση του μοντέλου.

Gaussian Smoothing: Εφαρμογή Gaussian kernel για μείωση θορύβου:

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \tag{Β΄.4}$$

Παράρτημα Γ΄

Οδηγίες Εγκατάστασης και Εκτέλεσης

Γ΄.1 Prerequisites

Απαιτήσεις συστήματος:

- **Python**: Έκδοση 3.8 ή νεότερη
- **Operating System**: Windows, macOS, ή Linux
- **Memory**: Τουλάχιστον 4GB RAM
- **Storage**: 30MB ελεύθερου χώρου
- **Internet**: Για λήψη δεδομένων από Alpha Vantage API

Γ΄.2 Installation Steps

Γ΄.2.1 Βήμα 1: Clone το Repository

Clone το Repository

```
git clone https://github.com/IBilba/stock-price-linear-regression.git
cd stock-price-linear-regression
```

Γ΄.2.2 Βήμα 2: Δημιουργία Virtual Environment

Windows:

Δημιουργία Virtual Environment

```
python -m venv venv
venv\Scripts\activate
```

macOS/Linux:

Δημιουργία Virtual Environment

```
python3 -m venv venv
source venv/bin/activate
```

Γ΄.2.3 Βήμα 3: Εγκατάσταση Dependencies

Εγκατάσταση Dependencies

```
pip install -r requirements.txt
```

Το `requirements.txt` περιλαμβάνει:

- numpy>=1.21.0
- pandas>=1.3.0
- scikit-learn>=1.0.0
- matplotlib>=3.4.0
- scipy>=1.7.0
- requests>=2.26.0
- python-dotenv>=0.19.0

Γ΄.2.4 Βήμα 4: Ρύθμιση Alpha Vantage API Key

Δημιουργήστε αρχείο `.env` στο root directory:

.env

```
api_key=your_api_key_here
```

Λάβετε δωρεάν API key από: Alpha Vantage

⚠ ΣΗΜΑΝΤΙΚΟ ΓΙΑ ΔΩΡΕΑΝ API KEYS / IMPORTANT FOR FREE API KEYS

Αν χρησιμοποιείτε **δωρεάν** API key, πρέπει να αλλάξετε την παράμετρο `outputsize=full` σε **`outputsize=compact`** στα αρχεία:
If using a **free** API key, you must change the parameter `outputsize=full` to **`outputsize=compact`** in the files:

- `step1_data_acquisition.py` (γραμμή / line ~78)
- `nflx_stock_prediction_complete_pipeline.ipynb` (κελί συλλογής δεδομένων / data acquisition cell)

Λόγος / Reason: Το `outputsize=full` απαιτεί premium API key. Το `compact` επιστρέφει τα τελευταία 100 data points.
`outputsize=full` requires a premium API key. `compact` returns the latest 100 data points.

Γ΄.3 Running Scripts vs Notebook

Γ΄.3.1 Εκτέλεση Python Scripts (Συνιστάται)

Εκτελέστε τα scripts με τη σωστή σειρά:

Εκτέλεση Python Scripts

```
# Βήμα 1: Συλλογή δεδομένων και εξομάλυνση
```

```
python step1_data_acquisition.py

# Βήμα 2: Feature engineering
python step2_feature_engineering.py

# Βήμα 3: Baseline linear regression
python step3_baseline_linear_regression.py

# Βήμα 4: Polynomial regression με L1/L2
python step4_polynomial_regression_regularization.py

# Βήμα 5: Dimensionality reduction
python step5_dimensionality_reduction.py

# Βήμα 6: Future predictions
python step6_future_predictions.py
```

Γ΄.3.2 Εκτέλεση Jupyter Notebook

Εναλλακτικά, χρησιμοποιήστε το integrated notebook:

Εκτέλεση Jupyter Notebook

```
jupyter notebook nflx_stock_prediction_complete_pipeline.ipynb
```

Το notebook περιέχει όλα τα βήματα σε ένα αρχείο με λεπτομερείς επεξηγήσεις.

Γ΄.4 Troubleshooting

Γ΄.4.1 Κοινά Προβλήματα

1. API Rate Limit Exceeded:

- Το Alpha Vantage free tier επιτρέπει 5 calls/minute, 500 calls/day
- Λύση: Περιμένετε 1 λεπτό μεταξύ των calls ή χρησιμοποιήστε cached data

2. Module Not Found Error:

Εγκατάσταση Dependencies

```
# Βεβαιωθείτε ότι το virtual environment είναι activated
pip list  \# Ελέγξτε εγκατεστημένα packages
pip install --upgrade -r requirements.txt
```

3. Memory Error:

- Μειώστε τον αριθμό των παράλληλων μοντέλων
- Κλείστε άλλες εφαρμογές
- Χρησιμοποιήστε batch processing

4. Matplotlib Display Issues:

Matplotlib Display Issues

```
# Για headless servers
import matplotlib
matplotlib.use('Agg')
```

Γ΄.4.2 Verification Tests

Ελέγξτε ότι όλα λειτουργούν σωστά:

Verification Tests

```
# Test imports
python -c "import numpy, pandas, sklearn; print('OK')"

# Test data files
ls data/nflx_monthly_*.csv

# Test feature files
ls features/features_*.npz

# Test model files
ls models/*.pkl
```